Zhiming Liu
Alan Wassyng (Eds.)

LNCS 7151

Foundations of Health Informatics Engineering and Systems

First International Symposium, FHIES 2011
Johannesburg, South Africa, August 2011
Revised Selected Papers

Springer

Zhiming Liu   Alan Wassyng (Eds.)

Foundations of Health Informatics Engineering and Systems

First International Symposium, FHIES 2011
Johannesburg, South Africa, August 29-30, 2011
Revised Selected Papers

Springer

Volume Editors

Zhiming Liu
United Nations University
International Institute for Software Technology
Casa Silva Mendes
Est. do Engenheiro Trigo No. 4
Macao, China
E-mail: lzm@iist.unu.edu

Alan Wassyng
McMaster University
Centre for Software Certification
1280 Main Street West, ITB 202
Hamilton, ON, Canada L8S 4K1
E-mail: wassyng@mcmaster.ca

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-32354-6 e-ISBN 978-3-642-32355-3
DOI 10.1007/978-3-642-32355-3
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012943388

CR Subject Classification (1998): J.3, D.2.4, H.4.1, H.2.7, D.2.11-12, I.6

LNCS Sublibrary: SL 2 – Programming and Software Engineering

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Information and communication technology plays an increasingly important enabling role in addressing the global challenges of healthcare, both in the developed and the developing world. Health care challenges are of concern for the United Nations, its peoples and member states. The use of software in medical devices has already raised issues in relation to safety and efficacy, both for manufacturers and regulators. One only has to look at the experience with infusion pumps in the USA, where thousands of serious incidents have been recorded. Health information systems raise issues of privacy, confidentiality, and even patient safety. The trend toward higher levels of interconnection between devices and with hospital information systems is leading to increased complexity and concomitant safety issues as evidenced by recent fatalities due to interconnection of radiation machines with hospital billing systems in the USA. Therefore, to capitalize on the potential of this technology to reshape healthcare services, and at the same time avoid harm to patients, will require focused research on sound and safe development techniques from software engineering, electronic engineering, computing science, information science, mathematics, and industrial engineering. The purpose of the new symposium series on Foundations of Health Information Engineering and Systems (FHIES) is to promote a nascent research area that aims to develop and apply theories and techniques in computing science and software engineering to modelling, building, and certifying software-based systems in the application domain of healthcare. Many of these systems are already regulated in various jurisdictions and many more of them will become regulated in the future. This symposium has created a forum for discussion of ideas and presentations in the following two areas:

1. Research results on how computational models, techniques, and tools of analysis and verification (including modelling notations, semantics, logics, techniques of model checking, runtime monitoring, and simulation) can be applied to problems of medical informatics
2. Application of foundational methods from software engineering in health informatics

The call for submissions attracted 23 submissions from around the world in a number of the areas mentioned in the call for papers. After the normal and thorough peer review process (papers were reviewed by three or four members of the Program Committee), 14 papers were accepted for presentation in the proceedings of the workshop. These papers were published as joint technical reports of the *United Nations University – International Institute for Software Technology* and the *McMaster Centre for Software Certification*. The symposium took place in South Africa in August 2011, was co-located with the International Colloquium on Theoretical Aspects of Computing, and succeeded beyond our expectation. The format we chose for the symposium included significant time for discussion, and this was highly appreciated by the participants. Authors had been informed of our intention to publish a post-proceedings volume, and after the completion of the symposium most authors decided to submit revised papers for these proceedings. These 13 papers were revised according to comments received at the symposium, and then went through an additional, rigorous review process. The papers cover a range of relevant and interesting topics in health informatics. There are several papers on workflow modelling and analysis, including formal modelling and analysis of medical protocols and workflows, applying model checking to analyze safety of workflows, and modelling of collaborative workflows. There are several papers on electronic medical records (EMRs) and medical information systems, including safety issues for EMRs, interoperability of health information systems and open architectures for health information systems in developing countries. There are papers on security and privacy in health information systems and even a paper on the application of model-driven development techniques to systematically develop software to support clinical trials in healthcare!

It is our hope and intention that this symposium will engender a long and productive series of meetings on foundational techniques in health informatics and experience of using these techniques in practice. Planning for the next meeting is well under way, and the second instantiation of FHIES will be co-located with Formal Methods 2012, to take place in August 2012 in Paris. We would like to thank our sponsors for their support throughout this venture: The United Nations University International Institute for Software Technology, The University of the Witwatersrand, and the McMaster Centre for Software Certification. We would also like to acknowledge the invaluable support of Tom Maibaum and Peter Haddawy, the General Co-chairs of FHIES 2011, the superb Program Committee, and Chris George and Johannes Faber for checking the final versions of the papers and compiling this volume.

February 2012

Zhiming Liu
Alan Wassyng

Organization

FHIES was organized jointly by the International Institute for Software Technology of the United Nations University (UNU-IIST) and the Centre for Software Certification of McMaster University (McSCert).

General Chairs

Peter Haddawy	UNU-IIST, Macao
Tom Maibaum	McSCert, Canada

Program Chairs

Zhiming Liu	UNU-IIST, Macao
Alan Wassyng	McSCert, Canada

Program Committee

Syed Mohamed Aljunid	UNU-IIGH, Malaysia
Sebastian Fischmeister	University of Waterloo, Canada
Peter Haddawy	UNU-IIST, Macao
Jozef Hooman	Embedded Systems Institute
Michaela Huhn	TU Clausthal, Germany
Mark Lawford	McSCert, Canada
Insup Lee	University of Pennsylvania, USA
Martin Leucker	University of Lübeck, Germany
Orlando Loques	Universidade Federal Fluminense, Brazil
Wendy Maccaull	St. Francis Xavier University, Canada
Tom Maibaum	McSCert, Canada
Dominique Méry	LORIA and Université Henri Poincaré Nancy 1, France
Jun Pang	University of Luxembourg
Anders Ravn	Aalborg University, Denmark
David Robertson	University of Edinburgh, UK
Martin Schäf	UNU-IIST, Macao
Lutz Schröder	DFKI Bremen and University of Bremen, Germany
Alexandre Sztajnberg	Universidade do Estado do Rio de Janeiro, Brazil
Umit Topaloglu	University of Arkansas for Medical Sciences, USA
Jens Weber	University of Victoria, Canada

Additional Reviewers

Table of Contents

Medical Protocol Diagnosis Using Formal Methods

Dominique Méry and Neeraj Kumar Singh

Université de Lorraine
LORIA, BP 70239, 54506 Vandoeuvre lès Nancy, France
{dominique.mery,singhnne}@loria.fr

Abstract. Clinical guidelines systematically assist practitioners to provide appropriate health care in specific clinical circumstances. Today, a significant number of guidelines and protocols are lacking in quality. Indeed, ambiguity and incompleteness are likely anomalies in medical practice. Our objective is to find anomalies and to improve the quality of medical protocols using well-known mathematical formal techniques, such as EVENT B. In this paper, we use the EVENT B modelling language to capture guidelines for their validation. Our main contributions are: to apply mathematical formal techniques to evaluate real-life medical protocols for quality improvement; to derive verification proofs for the protocol and properties according to medical experts; and to publicize the potential of this approach. An assessment of the proposed approach is given through a case study, relative to a real-life reference protocol (ECG interpretation), which covers a wide variety of protocol characteristics related to several heart diseases. We formalize the reference protocol, verify a set of interesting properties of the protocol and finally determine anomalies.

Keywords: Electrocardiogram (ECG), Medical protocol, Abstract model, EVENT B, Event-driven approach, Proof-based development, Refinement.

1 Introduction

A promising and challenging application area for the application of formal methods is clinical decision-making, as it is vital that clinical decisions should be sound. In fact, ensuring safety is the primary preoccupation of medical regulatory agencies. Medical guidelines are "systematically developed statements to assist practitioners and patients to determine appropriate health care for specific circumstances" [1,2]. Based on updated empirical evidence, medical protocols provide clinicians with health care treatment plans and facilitate the spreading of high-standard practices. In fact, adherence to protocols may reduce the costs of care by as much as 25% [2]. To realize their potential benefits, protocols must fulfil strong quality requirements. Medical bodies worldwide have made efforts in this direction; e.g. elaborating appraisal documents that take into account a variety of aspects of both protocols and their development processes. However, these initiatives are not sufficient, because they rely on informal methods and notations, which have no mathematical foundation.

We advocate a different approach: the quality improvement of medical protocols through formal methods. In this paper, we propose an approach for formalization and verification of any medical protocol for diagnosis purposes. Our work entails translating informal descriptions of a medical protocol into a more formal language, with the aim of analysing a set of properties. In addition to the advantages of formal verification, making these descriptions more formal can serve to expose problematic elements in the protocols. Currently, protocols are described using a combination of different formats, including text, flow diagrams and tables. These approaches typify informal processes and notations for analysing medical protocols, which are not sufficient for medical practice. As a result, medical guidelines and protocols[1] may contain ambiguous, incomplete or even inconsistent elements.

Formal methods have well-structured representation languages with clear and well-defined semantics, which can be used for taxonomy verification of clinical guidelines and medical protocols. The representation language represents guidelines and protocols explicitly in a non-ambiguous way. The process of verification using formal semantic representation of guidelines and protocols allows the determination of consistency and correctness. To model any medical protocol, we consider five basic objectives.

- To establish a unified theory and proper guidelines for analysing a medical protocol.
- To find ambiguity, incompleteness and inconsistency in the medical protocol.
- To determine requirements and metrics for certifiable assurance and safety.
- To build a comprehensive and integrated suite of tools for analysing medical protocols.
- To employ refinement-based formal development to achieve less error-prone models, easier specification of the medical protocol and reuse of such specification for further diagnosis.

Our approach is based on the EVENT B [3] modelling language, which is supported by the RODIN [4] platform, which integrates tools for proving and refining models. Here, we present an incremental proof-based development to model and verify such interdisciplinary requirements in EVENT B. Medical protocol models must be validated to ensure that they meet the requirements for medical protocols. Hence, validation must be carried out by both formal modelling and medical domain experts.

The refinement supported by the RODIN platform guarantees the preservation of safety properties. The main safety property is the detection of an actual disease under appropriate conditions. The behaviour of the final system is preserved by an abstract model as well as in the correctly refined models. The main idea is to start with a very abstract model of the (closed) target system under development. Details are gradually added to this first model by building a sequence of more concrete events. The relationship between two successive models in this sequence is *refinement* [3]. In the current work, we intend to explore problems related to the modelling of medical protocols. Moreover, an incremental development of the medical protocol model helps to discover any ambiguous, incomplete or even inconsistent elements in the medical protocol. Formal modelling and verification of medical protocols have been carried out as a case study to assess the feasibility of this approach. Throughout our case study of the electrocardiogram (ECG) interpretation protocol [5], we have shown formal specification and verification of the ECG interpretation protocols.

[1] "Guideline" and "protocol" are different terms. The term protocol is used to identify a specialized version of a guideline. In this paper, however, we do not distinguish them.

The outline of the rest of the paper is as follows. Section 2 describes related work. Section 3 explains the selection of a medical protocol for formalization. Section 4 presents the modelling framework. In Section 5, we explore the incremental proof-based formal development of the ECG interpretation protocol. The verification results are analysed by statistics relating to the proofs and lessons learned in Section 6. Finally, in Section 7, we conclude the paper and discuss future challenges.

2 Related Work

In recent years, many languages have been developed for representing medical guidelines and protocols using various levels of formality based on experts' requirements. Various protocol representation languages like Asbru [6,2], EON [7], PROforma [8] and others [9,10] have been used to represent a formal semantics of guidelines and medical protocols. Clinical guidelines are useful tools for providing some standardization and helping to improve protocols. A survey paper [11] presents benefits and comparisons through an analysis of the different kinds of systems used for clinical guidelines. This paper covers a wide scope of the related literature and tools, which have been collected from the medical informatics area.

One approach for improving guidelines and protocols is by evaluating the physician. Evaluation involves a scenario and evidence-based testing, which compares actions taken with the scenario. The actions are performed by physicians to handle particular patient cases using treatment plans that are prescribed by the guidelines [12]. When the results of the actions deviate, the evaluation process can either be focused on the explanation or provide some valuable feedback for improving guidelines and protocols [13]. An intention-based evaluation process is conducted by the evaluating physicians using both the patient data and the performed actions. These are then verified against the intentions reported in the guidelines.

Decision-table-based techniques for the verification and simplification of guidelines were presented by Shiffman et al. [14,15]. The basic idea behind this approach is to describe guidelines as condition/action statements: "If the antecedent circumstances exist, then perform the recommended actions" [15]. Completeness and consistency are the two main properties for verification when guidelines and protocols are expressed in terms of decision tables. Again, these properties are internal-coherence properties, whereas we focus on domain-specific properties.

Formal development of guidelines and protocols using clinical logic may be incomplete or inconsistent. This problem was tackled by Miller et al. [16]. "If 'if–then' rules are used as the representation language for guidelines, incompleteness means that there are combinations of clinically meaningful conditions to which the system (guideline) is not able to respond" [16]. The verification of rule-based clinical guidelines using semantic constraints was supported by the commander tool. This tool is able to identify clinical conditions where the rules are incomplete. Miller et al. [16] were able to find a number of missing rules in various case studies of guidelines and protocols.

Guideline enhancement was enabled through the adoption of advanced artificial intelligence techniques [17]. The authors proposed an approach for verification of the guidelines that was based on the integration of a computerized guideline management system with a model checker. They used the SPIN model checker [18,19] to execute and verify medical protocols or guidelines. A framework for authoring and verifying clinical guidelines was provided by Beatriz et al. [20]. The verification process for guidelines was based on the combined approach of Model-Driven Development (MDD) and model checking [19] to verify guidelines against semantic errors and inconsistencies. A UML [21,22] tool was used to model the guidelines, and a generated formal model was used as the input model for a model checker.

Jonathan et al. [23] proposed the application of formal methods through interactive verification to improve the quality of medical protocols or guidelines. They applied this technique to the management of jaundice in newborns based on guidelines from the American Academy of Pediatrics. This paper includes formalization of the jaundice protocol and verifies some interesting properties. Simon et al. [24] attempted to improve the same protocol using the modelling language Asbru, temporal logic for expressing the quality requirements, and model checking for proof and error detection.

Applying a formal approach to improve medical protocol is a major area of research that helps medical practitioners to improve the quality of patient care. Protocure [25] is a European project being conducted by five different institutions. The main objective of this project is to improve medical protocols through integration of formal methods. The main motivation for this project is to identify anomalies such as ambiguities and incompleteness in medical guidelines and protocols. At present, all medical protocols and guidelines are in text, flow diagrams and table formats, which are easily understandable by medical practitioners. However, they are incomplete and ambiguous because of the lack of formal semantics. The idea of using formal methods is to uncover the ambiguous, incomplete or even inconsistent parts of the protocols, by defining all the different descriptions more precisely using a formal language and thereby enabling verification. The researchers have mainly used the Asbru [2] language for protocol description and KIV for interactive verification [26].

Asbru [2,27] is a modelling language for describing medical protocols. The formal proof of the medical protocol is possible by using the KIV interactive theorem prover [26]. Guideline Markup Tool (GMT) [28] is an editor that helps in translating the guidelines into Asbru. An additional functionality of the tool is to define relations between the original protocol and its Asbru translation with a link macro [28]. The Asbru language is used for protocol description, and Asbru formalizations are translated into KIV. Asbru is considered a rigorous and semantically based language to support the tasks necessary for protocol-based care. It is called a rigorous language because of its formal semantics [27]. This rigorous quality makes Asbru suitable for an initial analysis; however, we advocate a progressive, incremental, proof-based approach to define a formal object from a list of expert-based requirements, and Asbru is not yet equipped with the refinement that appears to be the key tool for creating better links between expert requirements and formal objects. This does not mean that KIV is useless, but the focus should be on the methodology for producing formal models from informal or semiformal statements.

In this study, we have tried to model a medical protocol using only formal semantics and to check various anomalies. To overcome the existing problems [29,30] in developing medical protocols, we have used the general formal modelling notation of

EVENT B [3] for specifying a complex medical protocol related to diagnosis from ECG signals. The main objective in using the EVENT B modelling language is to model medical protocols using the refinement approach. Medical protocols are very complex, and a refinement approach is very helpful in modelling them, as it introduces the peculiarities of a protocol in an incremental way. This technique is used to model a medical protocol more rigorously based on formal mathematics, which helps to find anomalies and provides consistency and correctness of the medical protocol.

3 Selection of Medical Protocol

As the object of our study, we have selected ECG interpretation, which covers a wide range of protocol characteristics related to heart diseases. Medical guidelines and protocols differ from each other along several dimensions that relate to the contents of the protocols or to their form. General practitioners (GPs), nurses and a large group of people related to this domain[2] are the *most important target users* of the guidelines and protocols, and the main aspect of clinical practice is to facilitate *diagnosis* as well as to help in treatment. The medical guidelines and protocols used by general practitioners and nurses are also characterized by time dimensions: short-time-span protocols and long-time-span protocols. The forms of guidelines and protocols are related to their textual descriptions. Some protocols are represented in both textual form and *tables* and *flowcharts*.

The ECG interpretation protocol [31,32] aims at cardiologists as well as GPs and covers both diagnosis and treatment over a long period. The ECG interpretation protocol can be considered more precisely: it is used daily by cardiologists and it is included in the repository of the National Guideline Clearinghouse (NGC) and the American College of Cardiology/American Heart Association (ACC/AHA). The basic standard for inclusion in the NGC and ACC/AHA is that the guidelines and protocols contain well-structured meaningful information and systematically developed statements. The contents are produced under the supervision of medical specialty associations. They should also be based on the literature reviewed and revised within the last five years. Furthermore, the ECG interpretation protocol has been published in a peer-reviewed scientific journal. In summary, the chosen protocol covers different aspects while fulfilling high quality standards, which are good criteria for selection for our case study.

In the following sections, we will use the ECG interpretation protocol as the main example in our explanations, and we therefore give a brief description of this protocol.

3.1 Basic Overview of Electrocardiogram (ECG)

The electrocardiogram (ECG or EKG) [31] is a diagnostic tool that measures and records the electrical activity of the heart precisely in the form of signals. Clinicians can evaluate the conditions of a patient's heart from the ECG and perform further diagnosis. Analysis of these signals can be used for diagnosing a wide range of heart conditions and predict related diseases. ECG records are obtained by sampling the bioelectric currents sensed by several electrodes, known as leads.

[2] `http://www.guideline.gov/`

A typical one-cycle ECG tracing is shown in Fig. 1. All kinds of segments and intervals are represented in this ECG diagram. Depolarization and repolarization of the ventricular and atrial chambers are represented by deflections in the ECG signal. All these deflections are denoted in alphabetical order (P-QRS-T). the letter P indicates atrial depolarization, and the ventricular depolarization is represented by the QRS complex. The normal duration of the QRS-complex is 80–90 ms. Ventricular repolarization is represented by the T-wave. Atrial repolarization appears during the QRS complex and generates a very-low-amplitude signal that cannot be recovered from the normal ECG signal.

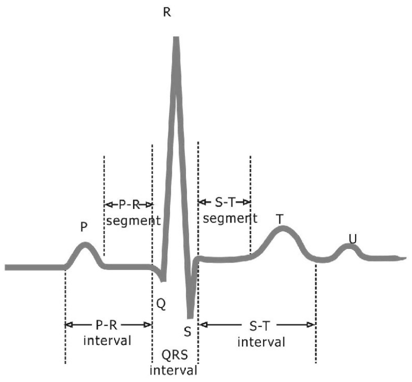


Fig. 1. One cycle of an ECG trace

In the ECG signal, several parameters are used to evaluate the condition of a patient's heart. The parameters are: PR-interval, P-wave, QRS duration, Q-wave, R-wave, ST-segment, T-wave, Axis, and QT-interval. All these parameters have several different characteristics that are used for diagnosis.

4 A Brief Introduction to the Modelling Framework

We summarize the concepts of the EVENT B modelling language developed by Abrial [3] and indicate the links with the tool called RODIN [4]. The modelling process deals with various languages, as seen by considering the *triptych*[3] of Bjoerner [33]: $\mathcal{D}, \mathcal{S} \longrightarrow \mathcal{R}$. Here, the domain $\mathcal{D}$ deals with properties, axioms, sets, constants, functions, relations, and theories. The system model $\mathcal{S}$ expresses a model or a refinement-based chain of models of the system. Finally, $\mathcal{R}$ expresses the requirements for the system being designed. Considering the EVENT B modelling language, we notice that the language can express *safety* properties, which are either *invariants* or *theorems* in a machine corresponding to the system. Recall that two main structures are available in EVENT B:

- Contexts express static information about the model;
- Machines express dynamic information about the model, invariants, safety properties, and events.

An EVENT B model is defined either as a context or as a machine. Bjoerner's triptych [33,34] $\mathcal{D}, \mathcal{S} \longrightarrow \mathcal{R}$ is translated as follows: $\mathcal{C}, \mathcal{M} \longrightarrow \mathcal{R}$, where $\mathcal{C}$ is a context, $\mathcal{M}$ is a machine and $\mathcal{R}$ are the requirements. The relation $\longrightarrow$ is defined to be a logical satisfaction relation with respect to an underlying logico-mathematical theory. The satisfaction relation is supported by the RODIN platform. A machine organizes events

[3] The term "triptych" covers the three phases of software development: domain description, requirements prescription and software design.

that modify state variables, and it uses static information defined in a context. These basic structure mechanisms are extended by the refinement mechanism, which provides a mechanism for relating an abstract model and a concrete model by adding new events or by adding new variables. This mechanism allows us to develop EVENT B models gradually and to validate each decision step using the proof tool. The refinement relationship should be expressed as follows: a model M is refined by a model P, when P simulates M. The final concrete model is close to the behaviour of the real system that is executing events using real source code. Note that our models can only express *safety* and *invariance* properties, which are state properties; the RODIN tool provides the required proof obligations (see the following subsections) for checking model correctness according to those properties. If we consider liveness properties, or, more generally, eventuality properties, we must state stronger properties over traces and extend the refinement relation by considering traces; this point is clearly addressed in any temporal framework such as (UNITY [35], TLA$^+$ [36], or Action Systems [37], but the tools are not as well developed as those in RODIN. Model checking tools can also be used when dealing with temporal properties, but it appears that the technique is applicable only to finite models. RODIN provides a modelling framework for discrete models but not for continuous or hybrid models; it is less powerful than general temporal (real-time) models that can be analysed by UPPAAL model checking, for instance. Now, we give details on the definition of events, refinement and guidelines for developing complex system models.

4.1 Modelling Actions over States

EVENT B [3] is based on the B notation. It extends the methodological scope of basic concepts to take into account the idea of *formal reactive models*. Briefly, a formal reactive model is characterized by a (finite) list x of *state variables*, possibly modified by a (finite) list of *events*, where an invariant $I(x)$ states properties that must always be satisfied by the variables x and *maintained* by activations of the events. In the following, we summarize the definitions and principles of formal models and explain how they can be managed by tools [4].

Generalized substitutions are borrowed from the B notation, which express changes in the value of state variables. An event has three main parts: a list of local parameters, a guard and a relation over values denoted *pre*-values and *post*-values of variables. The most common event representation is (ANY t WHERE $G(t,x)$ THEN $x : |(R(x,x',t))$ END). The *before–after* predicate $BA(e)(x,x')$ associated with each event describes the event as a logical predicate for expressing the relationship linking values of the state variables just before (x) and just after (x'), the *execution* of event e. The form is semantically equivalent to $\exists t \cdot (G(t,x) \;\wedge\; R(x,x',t))$. An event is equivalent to a *reactive action* waiting for a condition called a guard and modifying state variables according to the relation between the pre-value x of x and the post-value x' of x.

Table 1. EVENT B proof obligations

PROOF OBLIGATIONS
– (INV1) $Init(x) \Rightarrow I(x)$
– (INV2) $I(x) \wedge BA(e)(x,x') \Rightarrow I(x')$
– (FIS) $I(x) \wedge \mathsf{grd}(e)(x) \Rightarrow \exists y.BA(e)(x,y)$

Proof obligations (INV 1 and INV 2) are produced by the RODIN tool [4] from events to state that an invariant condition $I(x)$ is preserved. Their general form follows immediately from the definition of the before–after predicate $BA(e)(x,x')$ of each event e (see Table 1). Note that it follows from the two guarded forms of the events that this obligation is trivially discharged when the guard of the event is false. Whenever this is the case, the event is said to be *disabled*. The proof obligation FIS expresses the feasibility of the event e with respect to the invariant I. By proving feasibility, we prove that $BA(e)(x,y)$ provides an after-state whenever $grd(e)(x)$ holds. This means that the guard indeed represents the enabling condition of the event.

The intention of specifying a guard of an event is that the event may always occur when the given guard is true. There is, however, some interaction between guards and nondeterministic assignments: $x : |BA(e)(x,x')$, read as *x receives the value v satisfying BA(e)(x, v), when the value v exists*. The predicate $BA(e)(x,x')$ of an action $x : |BA(e)(x,x')$ may be not satisfiable; this means that no value satisfying the relation exists. This case shows a violation of the event feasibility proof obligation. We say that an assignment is feasible if there is an after-state satisfying the corresponding before–after predicate. For each event, its feasibility must be proved. Note that for deterministic assignments, the proof of feasibility is trivial. Note also that the feasibility of the initialization of a machine yields the existence of an initial state of the machine. It is not necessary to require an extra initialization.

4.2 Model Refinement

The refinement of a formal model allows us to enrich the model via a *step-by-step* approach and is the foundation of our correct-by-construction approach [38]. Refinement provides a way to strengthen invariants and to add details to a model. It is also used to transform an abstract model to a more concrete version by modifying the state description. This is done by extending the list of state variables (and possibly suppressing some of them), by refining each abstract event to a set of possible concrete versions, and by adding new events. The abstract (x) and concrete (y) state variables are linked by means of a *glueing invariant* $J(x,y)$. A number of proof obligations ensure that: (1) each abstract event is correctly refined by its corresponding concrete version; (2) each new event refines $skip$; (3) no new event takes control for ever; and (4) relative deadlock freedom is preserved. Details of the formulation of these proofs follows.

We suppose that an abstract model AM with variables x and invariant $I(x)$ is refined by a concrete model CM with variables y and glueing invariant $J(x,y)$. Event e is in abstract model AM, and event f is in the concrete model CM. Event f refines event e. $BA(e)(x,x')$ and $BA(f)(y,y')$ are predicates of events e and f respectively. We must prove the following statement, corresponding to proof obligation (1).

$$\boxed{I(x) \;\wedge\; J(x,y) \;\wedge\; BA(f)(y,y') \;\Rightarrow\; \exists x' \cdot (BA(e)(x,x') \;\wedge\; J(x',y'))}$$

The new events introduced in a refinement step can be viewed as hidden events not visible to the environment of the system and are thus outside the control of the environment. In EVENT B, requiring a new event to refine $skip$ means that the effect of the new event is not observable in the abstract model. Any number of executions of an internal action may occur between executions of a visible action. Now, proof obligation (2) states that $BA(f)(y, y')$ must refine $skip$ $(x' = x)$, generating the following simple statement to prove (2).

$$\boxed{I(x) \;\wedge\; J(x,y) \;\wedge\; BA(f)(y,y') \;\Rightarrow\; J(x,y')}$$

In refining a model, an existing event can be refined by strengthening the guard and/or the before–after predicate (effectively reducing the degree of nondeterminism), or a new event can be added to refine the $skip$ event. The feasibility condition is crucial for avoiding possible states that have no successor, such as division by zero. Furthermore, this refinement guarantees that the set of traces of the refined model contains (up to stuttering) the traces of the resulting model. The refinement of an event e by an event f means that the event f simulates the event e.

The EVENT B modelling language is supported by the RODIN platform [4] and has been introduced in publications [3] describing many case studies and discussions about the language itself and the foundations of the EVENT B approach. The language of *generalized substitutions* is very rich, enabling the expression of any relation between states in a set-theoretical context. The expressive power of the language leads to a requirement for help in writing relational specifications, which is why we should provide guidelines for assisting the development of EVENT B models. The EVENT B modelling language is supported by the Atelier B [39] environment and by the RODIN platform [4]. Both environments provide facilities for editing machines, refinements, contexts and projects, for generating proof obligations corresponding to a given property, for proving proof obligations in an automatic or/and interactive process and for animating models. The internal prover is shared by the two environments, and hints are generated by the prover interface to assist with the interactive proofs. These tools are based on logical and semantic concepts of EVENT B models (machines, contexts, refinement), and our methodology for modelling medical protocol or guidelines can be built from them.

5 Formal Development of the ECG Interpretation

5.1 Abstract Model: Assessing Rhythm and Rate

We begin by defining the EVENT B context. The context uses sets and constants to define axioms and theorems, which represent the logical theory of the system. The logical theory lists the static properties of constants related to the system and provides an axiomatization of the systems environment. Context can be extended by other contexts and

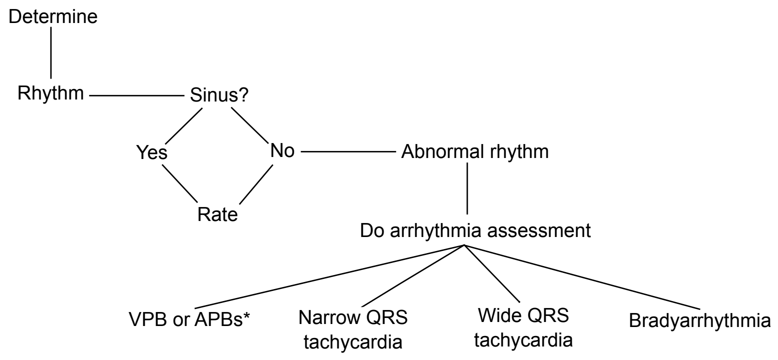


Fig. 2. Basic diagram describing assessing rhythm and rate [31]

referenced by a set of machines, while a machine can be refined by other machines. Figure 2 presents a basic diagram of ECG analysis at an abstract level according to the standard procedure of the ECG protocol analysis. In the context, we define constants *LEADS*, *HState* and *YesNoState*, which are related to the enumerated set of ECG leads, normal and abnormal states of the heart and yes–no states, respectively. These constants are extracted from the ECG interpretation protocol [31]. A set of leads is represented as *LEADS* = {*I, II, III, aVR, aVL, aVF, V1, V2, V3, V4, V5, V6*}. Normal and abnormal states of the heart are represented by *HState* = {*OK, KO*}, and yes–no states are represented by *YesNoState* = { *Yes, No* }.

Figure 3 depicts an incremental formal development of the ECG interpretation protocol. Every refinement level introduces *diagnosis* criteria for different components of the ECG signal, and each new criterion helps to analyse for a particular set of diseases. Figure 4 shows an abstract representation of a *diagnostic-based* system development, where the root node (top circle in Fig. 4) represents a set of conditions for testing for any particular disease abstractly. The possible abstract outcomes of diagnosis criteria are in the form of OK and KO, which are

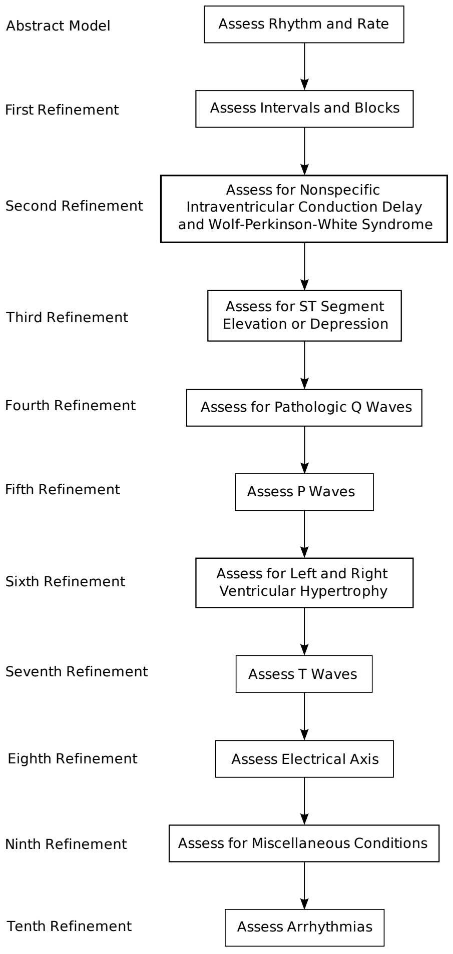

Abstract Model
Assess Rhythm and Rate
First Refinement
Assess Intervals and Blocks
Second Refinement
Assess for Nonspecific Intraventricular Conduction Delay and Wolf-Perkinson-White Syndrome
Third Refinement
Assess for ST Segment Elevation or Depression
Fourth Refinement
Assess for Pathologic Q Waves
Fifth Refinement
Assess P Waves
Sixth Refinement
Assess for Left and Right Ventricular Hypertrophy
Seventh Refinement
Assess T Waves
Eighth Refinement
Assess Electrical Axis
Ninth Refinement
Assess for Miscellaneous Conditions
Tenth Refinement
Assess Arrhythmias

Fig. 3. Refinements of the ECG protocol

represented by two branches. KO indicates that the diagnosis criteria have found some conditions for further testing, while OK indicates the absence of any disease. Dashed lines for circles and arrows represent the next level of refinement for further analysing for particular diseases according to the guidelines and protocol. Our abstract EVENT B model of the ECG interpretation protocol assesses *rhythm* and *heart rate* to distinguish between normal and abnormal heart conditions (see Fig. 2). The specification consists of just three state variables ($inv1 - inv3$): *Sinus*, *Heart_Rate* and *Heart_State*. The last two of these variables are introduced to show heart rate limit and heart states. Invariants ($inv4 - inv8$) represent the set of functions (*RR_Int_equidistant*, *PP_Int_equidistant*, *P_Positive*, *PP_Interval* and *RR_Interval*).

The *P_Positive* function is used to show the positive visualization of the P-waves. The *PP_Interval* and *RR_Interval* functions are used to calculate PP and RR intervals. All invariants ($inv9 - inv14$) represent safety properties, which are used to verify the required conditions for the ECG interpretation protocol based on analysis of the signal features. All these invariants are generated from the ECG interpretation protocol and extracted from the requirement documents with the help of medical experts. The invariant ($inv9$) states that if positive visualization of the P-wave is $FALSE$, then there is no sinus rhythm. According to the clinical document, lead II is best for the visualization of P-waves to determine the presence of sinus rhythm.

Invariant ($inv10$) states that if PP intervals (*PP_Int_equidistant*) or RR intervals (*RR_Int_equidistant*) are not equidistant ($FALSE$), or RR intervals (*RR_Interval*) and PP intervals (*PP_Interval*) are not equivalent in all leads (II, V1, V2), or positive visualization of P-wave in lead II is $FALSE$, then there is no sinus rhythm.

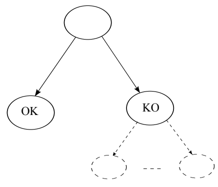


Fig. 4. Abstract representation

Three significant events *Rhythm_test_TRUE*, *Rhythm_test_FALSE* and *Rhythm_test_TRUE_abRate* are introduced in the abstract model. The *Rhythm_test_TRUE* represents successful ECG testing and sinus rhythm found *Yes* and heart state is *OK*. The next event, *Rhythm_test_FALSE*, represents successful ECG testing and found sinus rhythm is *No* and heart state is *KO*. The third event, *Rhythm_test_TRUE_abRate*, represents successful ECG testing and sinus rhythm found is *Yes* and heart state is *KO* because of abnormal heart rate (see Fig. 2).

$$
\begin{aligned}
&inv1 : Sinus \in YesNoState\\
&inv2 : Heart\_Rate \in 1 \,..\, 300\\
&inv3 : Heart\_State \in HState\\
&inv4 : RR\_Int\_equidistant \in LEADS \rightarrow BOOL\\
&inv5 : PP\_Int\_equidistant \in LEADS \rightarrow BOOL\\
&inv6 : P\_Positive \in LEADS \rightarrow BOOL\\
&inv7 : PP\_Interval \in LEADS \rightarrow \mathbb{N}\\
&inv8 : RR\_Interval \in LEADS \rightarrow \mathbb{N}\\
&inv9 : P\_Positive(II) = FALSE \Rightarrow Sinus = No\\
&inv10 : ((\forall l \cdot l \in \{II, V1, V2\} \Rightarrow\\
&\qquad PP\_Int\_equidistant(l) = FALSE \lor\\
&\qquad RR\_Int\_equidistant(l) = FALSE \lor\\
&\qquad RR\_Interval(l) \neq PP\_Interval(l))\\
&\qquad \lor\\
&\qquad P\_Positive(II) = FALSE) \Rightarrow Sinus = No\\
&inv11 : Sinus = Yes \Rightarrow ((\exists l \cdot l \in \{II, V1, V2\} \land\\
&\qquad PP\_Int\_equidistant(l) = TRUE \land\\
&\qquad RR\_Int\_equidistant(l) = TRUE \land\\
&\qquad RR\_Interval(l) = PP\_Interval(l))\\
&\qquad \land\\
&\qquad P\_Positive(II) = TRUE)\\
&inv12 : Heart\_Rate \in 60 \,..\, 100 \land Sinus = Yes \Rightarrow\\
&\qquad Heart\_State = OK\\
&inv13 : Heart\_Rate \in 1 \,..\, 300 \setminus 60 \,..\, 100 \land Sinus = Yes\\
&\qquad \Rightarrow Heart\_State = KO\\
&inv14 : Heart\_Rate \in 60 \,..\, 100 \land Sinus = No \Rightarrow\\
&\qquad Heart\_State = KO
\end{aligned}
$$

```
EVENT Rhythm_test_TRUE
  ANY $rate$
  WHEN
    grd1 : $(\exists l \cdot l \in \{II, V1, V2\} \land$
            $PP\_Int\_equidistant(l) = TRUE \land$
            $RR\_Int\_equidistant(l) = TRUE \land$
            $RR\_Interval(l) = PP\_Interval(l)) \land$
            $P\_Positive(II) = TRUE$
    grd2 : $rate \in 60 .. 100$
  THEN
    act1 : $Sinus := Yes$
    act2 : $Heart\_Rate := rate$
    act3 : $Heart\_State := OK$
END
```

```
EVENT Rhythm_test_FALSE
  ANY $rate$
  WHEN
    grd1 : $(\forall l \cdot l \in \{II, V1, V2\} \Rightarrow$
            $PP\_Int\_equidistant(l) = FALSE$
            $\lor RR\_Int\_equidistant(l) = FALSE \lor$
            $RR\_Interval(l) \neq PP\_Interval(l)) \lor$
            $P\_Positive(II) = FALSE$
    grd2 : $rate \in 1 .. 300$
  THEN
    act1 : $Sinus := No$
    act2 : $Heart\_Rate := rate$
    act3 : $Heart\_State := KO$
END
```

In the abstract model, we have seen that the sinus rhythm and heart rate are introduced for the ECG interpretation in a single atomic step. This provides a clear and simple specification of the essence of the basic ECG interpretation protocol and predicts the heart state (OK or KO). However, in the real protocol, the ECG interpretation and the heart state prediction are not atomic. Instead, they are part of many diagnoses to find the various heart diseases.

```
EVENT Rhythm_test_TRUE_abRate
  ANY $rate$
  WHEN
    grd1 : $(\exists l \cdot l \in \{II, V1, V2\} \land$
            $PP\_Int\_equidistant(l) = TRUE \land$
            $RR\_Int\_equidistant(l) = TRUE \land$
            $RR\_Interval(l) = PP\_Interval(l)) \land$
            $P\_Positive(II) = TRUE$
    grd2 : $rate \in 1 .. 300 \setminus 60 .. 100$
  THEN
    act1 : $Sinus := Yes$
    act2 : $Heart\_Rate := rate$
    act3 : $Heart\_State := KO$
END
```

First Refinement: Assess Intervals and Blocks. In an abnormal ECG signal, all ECG features vary according to the symptoms of the heart diseases. We will formalize the ECG interpretation protocol using an incremental approach, in which we determine all features of the ECG signal. This first level of refinement determines the PR and QRS intervals for the ECG interpretation. These intervals classify different kinds of heart disease.

Invariants ($inv1 - inv3$) represent a set of new introduced variables in the refinement for expressing formalization of the ECG interpretation protocol. These variables are *PR_Int*, *Disease_step2*, and *QRS_Int*. Other variables (*M_Shape_Complex*, *Slurred_S*, *Notched_R*, *Small_R_QS* and *Slurred_S_duration*) are introduced as total functions in invariants ($inv4 - inv8$), where total functions are mappings from leads (LEADS) to $BOOL$ and $\mathbb{N}_1$, respectively. Function *M_Shape_Complex* returns the existence of an M-shape complex from the ECG signals in form of $TRUE$ and $FALSE$. The function *Slurred_S* represents a slurred S-wave, the function *Notched_R* represents a notched R-wave, and the function *Small_R_QS* represents small R or QS waves, in boolean form. The function *Slurred_S_duration* is used to calculate the duration of the slurred S-wave. The set of invariants ($inv9 - inv14$) represents safety properties to validate formal representation of the ECG interpretation protocol. All these properties are derived from the original protocol to verify the correctness and consistency of the system. They were formulated by logic experts and cardiologists according to the original protocols. The main advantage of this technique is that if any property is not verified in the model, then it helps to find anomalies or to find missing parts of the model such as required conditions and parameters.

Invariants ($inv9 - inv13$) represent an abnormal state of the heart (KO) because of finding a disease, and they also represent an unsatisfiable condition for features of the

ECG signal in the formal diagnosis process. The last invariant ($inv14$) represents all the required properties for a normal heart. It states that if the heart rate is between 60 and 100 bpm, sinus rhythm is Yes, PR interval is less than or equal to 200 ms and QRS interval is less than 120 ms, then the heart state is OK.

To express the formal logic for a new set of diagnoses for the ECG signal, we have introduced three events: *PR_test*, *QRS_Test_LBBB* and *QRS_Test_RBBB*. From the *PR_Test* intervals, we can deduce that if the PR intervals are abnormal ($>$ 200 ms), we should consider first-degree atrioventricular (AV) block. *QRS_Test_LBBB* and *QRS_Test_RBBB* are used to assess the QRS duration for bundle branch block. If it is $\geq$ 120 ms, bundle branch block is present. Understanding the genesis of the QRS complex is an essential step and clarifies the ECG manifestations of bundle branch blocks [31]. We have formalized the basic criteria to distinguish between RBBB and LBBB in the diagnosis process. The basic descriptions of RBBB and LBBB are given as follows.

$inv1 : PR\_Int \in 120 \,..\, 250$
$inv2 : Disease\_step2 \in Disease\_Codes\_Step2$
$inv3 : QRS\_Int \in 50 \,..\, 150$
$inv4 : M\_Shape\_Complex \in LEADS \rightarrow BOOL$
$inv5 : Slurred\_S \in LEADS \rightarrow BOOL$
$inv6 : Notched\_R \in LEADS \rightarrow BOOL$
$inv7 : Small\_R\_QS \in LEADS \rightarrow BOOL$
$inv8 : Slurred\_S\_duration \in LEADS \rightarrow \mathbb{N}_1$
$inv9 : Sinus = Yes \land PR\_Int > 200 \land Disease\_step2 =$
$\quad First\_degree\_AV\_Block \Rightarrow Heart\_State = KO$
$inv10 : Sinus = Yes \land QRS\_Int \geq 120 \land$
$\quad Disease\_step2 \in \{LBBB, RBBB\} \Rightarrow Heart\_State = KO$
$inv11 : Sinus = Yes \land Disease\_step2 = First\_degree\_AV\_Block$
$\quad \Rightarrow Heart\_State = KO$
$inv12 : Sinus = Yes \land Disease\_step2 = LBBB \Rightarrow$
$\quad Heart\_State = KO$
$inv13 : Sinus = Yes \land Disease\_step2 = RBBB \Rightarrow$
$\quad Heart\_State = KO$
$inv14 : Heart\_Rate \in 60 \,..\, 100 \land Sinus = Yes \land PR\_Int \leq 200$
$\quad \land QRS\_Int < 120 \Rightarrow Heart\_State = OK$

Right Bundle Branch Block (RBBB)

– QRS duration $\geq$ 120 ms.
– M-shaped complex in leads V1 and V2.
– Slurred S-wave in leads 1, V5, V6; and an S-wave that is of greater amplitude than the preceding R-wave.

Left Bundle Branch Block (LBBB)

– QRS duration $\geq$ 120 ms.
– A small R- or QS-wave in leads V1 and V2.
– A notched R-wave in leads 1, V5, and V6.

Because of space limitations, we cannot show the complete formal representation of introduced events: for details see [5].

5.2 Overview of the Full Refinement Chain

So far, we have described our abstract model of the ECG interpretation protocol. Every level of refinement introduces a new context file for adding static properties to the system and a list of heart diseases. Each refinement level introduces a new set of diagnosis criteria to test the ECG signals. Rather than presenting the remaining refinement stages in similar detail (see in Section 5.1), we will present a sufficient overview of them to help the reader understand the rationale of each stage in formalizing the ECG interpretation protocol. All these refinements correspond to the standard analysis steps of the ECG protocol [31]. To find more detailed information see the technical report [5].

Second Refinement: Assess for Nonspecific Intraventricular Conduction Delay and Wolff–Parkinson–White Syndrome. This level of refinement of the ECG interpretation assesses for non-specific intraventricular conduction delay (IVCD) and Wolff–Parkinson–White (WPW) syndrome. WPW syndrome may mimic an inferior MI (see in further refinements). If neither WPW syndrome, nor RBBB, nor LBBB is present, interpret as non-specific intraventricular conduction delay (IVCD) and assess for the presence of electronic pacing using an artificial pacemaker.

Third Refinement: Assess for ST-Segment Elevation or Depression. This refinement provides the criteria for ST-segment assessment. To assess ST-segment elevation or depression, we have formalized the textual criteria given in [31]. This refinement advises scrutiny of the ST-segment before assessment of T-waves, electrical axis, QT interval, and hypertrophy because the diagnosis of acute MI or ischaemia is vital and depends on careful assessment of the ST-segment. Assessments of ST-segments are very complex and ambiguous. Therefore, we have formalized this part through careful cross-reading of many reliable sources such as literature and suggestions of medical experts.

Fourth Refinement: Assess for Pathologic Q-Wave. This refinement introduces new guidelines for interpreting the Q-wave feature of the ECG signal for diagnosis of diseases related to the assessment of the Q-wave and R-wave.

Fifth Refinement: P-Wave. This refinement level introduces a criterion for assessing the P-wave in the ECG signal for abnormalities, including atrial hypertrophy.

Sixth Refinement: Assess for Left and Right Ventricular Hypertrophy. Left ventricular Hypertrophy (LVH) and Right Ventricular Hypertrophy (RVH) are assessed by this refinement. The criteria for LVH and RVH are not applicable if any bundle branch block is present. Thus, it is essential to exclude LBBB and RBBB early in the interpretive sequences as we did in refinements 2 and 3.

Seventh Refinement: Assess T-Wave. This refinement is used to assess the pattern of T-wave changes in 12-lead ECG signals. T-wave changes are usually non-specific. The T-wave inversion associated with the ST-segment depression or elevation indicates myocardial ischaemia.

Eighth Refinement: Assess Electrical Axis. After finding all the information previously assessed for the abnormal ECG, it is also essential to check the electrical axis using two simple clues: First, if leads I and aVF are upright, the axis is normal; and second, if the axis is perpendicular to the lead with the smallest or most equiphasic QRS deflection. Left-axis deviation and the commonly associated left anterior fascicular block are visible in the ECG signal.

Ninth Refinement: Assess for Miscellaneous Conditions. There are many heart diseases, and it is very difficult to predict everything. Additional conditions make the interpretation more and more ambiguous. This refinement level incorporates multiple miscellaneous conditions about the ECG interpretation. The following conditions are given for miscellaneous conditions: If electronic pacing is confirmed using a pacemaker, usually no other diagnosis can be made from the ECG; and assessment of the QT intervals.

Tenth Refinement: Assess Arrhythmias. This is the final refinement of the ECG interpretation of the system. In this refinement, we introduce different kinds of tachyarrhythmias and give the protocols for assessment for narrow complex tachycardia and wide complex tachycardia.

We have given here only summaries of each refinement in the form of very basic descriptions of the ECG interpretation protocol using the incremental refinement-based approach and have omitted detailed formalization of events and proof details because of limited space. To find more detailed information with developed formal model of ECG interpretation protocol, see the technical report [5].

6 Statistical Analysis and Lessons Learned

6.1 Statistical Analysis

Table 2 shows statistics for the ECG interpretation protocol using the refinement approach. In the table, the POs column shows the total number of proof obligations generated for each level. The interactive POs column shows the number of those proof obligations that were proved interactively. The remaining proof obligations were proved completely automatically by the prover. The complete ECG interpretation protocol results in 599 (100%) proof obligations, of which 343 (58%) were proved automatically by the RODIN tool. The remaining 256 (42%) proof obligations were proved interactively using the RODIN tool. Therefore, all the proofs are discharged either completely automatically or interactively for all the refinement levels. All these proofs are complicated, either because of the complexity of the formal expressions that are proved by the *do case* or because of the finiteness constraints on the set of leads. Most importantly, we have devoted five months to developing this complex protocol with the help of medical experts. Some POs are trivial, while others are difficult and require some time to prove. We required only a total of three months to discharge all POs. In EVENT-B models, many proof obligations are generated because of the introduction of new

Table 2. Proof Statistics

Model	Total number of POs	Automatic Proof	Interactive Proof
Abstract Model	41	33 (80%)	8 (20%)
First Refinement	61	54 (88%)	7 (12%)
Second Refinement	41	38 (92%)	3 (8%)
Third Refinement	51	36 (70%)	15 (30%)
Fourth Refinement	60	35 (58%)	25 (42%)
Fifth Refinement	43	22 (51%)	21 (49%)
Sixth Refinement	38	14 (36%)	24 (64%)
Seventh Refinement	124	29 (23%)	95 (77%)
Eighth Refinement	52	30 (57%)	22 (43%)
Ninth Refinement	21	9 (42%)	12 (52%)
Tenth Refinement	67	43 (64%)	24 (36%)
Total	599	343 (58%)	256 (42%)

criteria for disease testing and their parameters. The main interactive steps involved instantiations for the total functions of the different features of the ECG interpretation in every level of refinement. To guarantee the correctness of the system, we have established various invariants in the stepwise refinement. Most of the invariants were introduced for checking for abnormalities of the ECG signal. By detecting any abnormal criteria, we demonstrate the presence of a particular disease or set of diseases in the heart. A set of diseases is distinguished in subsequent levels of refinement. It should be noted that some of the manual proofs were not difficult and were achieved with the help of the *do case* operation. Guards of some events are very complex, so to prove invariants and theorems, we simplified the guards using the *do case*. The stepwise refinement of the ECG medical protocol helps to achieve the high proportion of automatic proofs.

6.2 Lessons Learned

The task of modelling a medical protocol in EVENT B has required a significant effort. It is a typical knowledge engineering task, in which the knowledge in the original document is transformed into the EVENT B notation, which provides a significant hierarchical structure for analysing the medical protocol, which may be used in the improvement of the original protocol. The most important contribution is the refinement-based formal development of the medical protocol. The developed formal model is proved and verified according to the given protocol properties as discussed in the formal development. Furthermore, the EVENT B formalization has served to disambiguate unclear statements in the original document that were found in the modelling stage: a number of ambiguities and repetitive diagnosis problems with the original document were uncovered and resolved by refining the formal specification of the medical protocol in EVENT B. The formal model can help to restructure the original document of guidelines and protocols.

The EVENT B ECG interpretation protocol specification is much longer than the original text of the ECG interpretation protocol. The complete formal specification of the ECG interpretation protocol in EVENT B contains more than 200 pages [5]. The verification attempts have served to clarify any remaining problems in the original ECG interpretation protocol document. More importantly, we have shown that it is possible in practice to analyse systematically whether a protocol formalized in EVENT B complies with certain medically relevant properties.

More importantly, numerous anomalies became apparent during the EVENT B modelling of the medical protocol. Here, we have used term anomaly to refer to any issues in the original ECG interpretation protocol that could not be represented satisfactorily. Some of the anomalies are described below. We discovered more than a dozen anomalies during our protocol modelling and verification process. We have grouped all the anomalies into three well-known general categories: ambiguities, inconsistencies and incompleteness. Here, we have given only a few examples from the list of anomalies to show the kinds of anomalies discovered.

Ambiguities. Ambiguity is a well-known anomaly in the area of formal representation, and it can be very hard to interpret. For instance, a problem we encountered while modelling the ECG interpretation protocol was to determine whether or not the terms "ST-depression" and "ST-elevation" had the same meaning. These terms are used in the ECG interpretation original protocol but not defined elsewhere. Similarly, what is the difference between “ischaemia”, “Definite ischaemia”, “probable ischaemia” and “likely ischaemia”?

In the ECG interpretation, there are 12 ECG signals, but in many places, the original document does not clarify in which lead the particular property should hold.

Inconsistencies. Inconsistencies always give conflicting results or different decisions on the same patient data. The problems derived from inconsistent elements are very serious and as such must be avoided during development. The ECG interpretation protocol presents several inconsistencies. For instance, we found an inconsistency in the form of applicable conditions in the ECG protocol. It expresses that some conditions are applicable to both “male” and “female” subjects under certain circumstances. However, elsewhere in the protocol, an action advises that these conditions of the protocol are not applicable to “female” subjects.

Incompleteness. Incompleteness refers to either missing pieces of information or insufficient information in the original document. In either case, incompleteness hinders a correct interpretation of guidelines and protocols. For example, the original protocol contains “normal variant” factors to be considered when assessing the T-wave. However, exactly what this term means is not specified in the protocol. Because of insufficient information for a “normal variant”, we provided a class of disease for further analysis in the system.

7 Conclusions and Future Challenges

7.1 Conclusions

Refinement is a key concept in developing complex systems, because it starts from a very abstract model and incrementally adds new details to the set of requirements. We have outlined an incremental refinement-based approach for formalizing medical protocols using the RODIN tools. The approach we have taken is not specific to EVENT B. We believe a similar approach could be taken using other state-based notations such as ASM, TLA$^+$ or Z. The RODIN proof tool was used to generate the hundreds of proof obligations and to discharge those obligations automatically or interactively. In summary, some key lessons are that incremental development with small refinement steps, appropriate abstractions at each level and powerful tool support are all invaluable in formal developments.

In this paper, we have shown the formal representation of the medical protocol and verified it. This verified model is not only feasible but also useful for improving the existing medical protocol. We have fully formalized the real-world medical protocol (ECG interpretation) in an incremental refinement-based formalization process, and we have used proof tools to analyse systematically whether the formalization complies with medically relevant protocol properties. The formal verification process discovered a number of anomalies that are discussed in the previous section. Through this process, we have obtained the following concrete results.

- A formal specification language EVENT B was used to model the real-life medical protocol for ECG interpretation.

- The ECG interpretation protocol was formalized in EVENT B. The interpretation used in our study was developed in an incremental way. Each proven refinement level of the formal model of the protocol represents feasibility and correctness.
- In our formal verification process of the ECG interpretation, we obtained a list of anomalies.
- Our ECG interpretation protocol was proved using the RODIN tool. The generated proof obligations and proofs show that formal verification of the ECG interpretation protocol is feasible.
- The original ECG protocol is based on a hierarchy, where some diagnoses are repeated in multiple branches (see Section [31]). We have discovered an optimized hierarchical structure for the ECG interpretation that efficiently uses the incremental refinement approach, which may help practitioners to diagnose more efficiently than the old techniques, and the hierarchical structure obtained was verified by medical experts.

Our objective behind this work is that if a medical protocol is developed under particular circumstances to handle a set of specific properties according to the medical experts, formal verification can also show whether the protocol actually complies with them. To our knowledge, this is the first attempt to verify a medical protocol with mathematical rigour using a generalized formal modelling tool. The main objective of this approach was to test the correctness and consistency of the medical protocol using refinement-based incremental development. This approach not only may help with diagnoses but also may be applicable to many other categories (such as treatment, management, prevention, counselling, and evaluation)[4] related to the medical protocols.

7.2 Future Challenges

As we have seen, several techniques like modelling, formalizing and verifying are useful for finding anomalies in medical protocols. Some undecided issues that remain are: what kind of tools are applicable for finding anomalies; which types of techniques are required and when should they be applied; would it be possible to use verification and validation only in the critical parts of the guideline; and how can we identify these critical parts?

Fast changes in medical protocols (within two to three years) are considered a major problem[2]. After discussion with medical experts, we have found that it is an open problem in this area. However, in the last few years, medical scientific bodies have been updating current guidelines and protocols online, and providing up-to-date information to all users.

We have used the EVENT B modelling language for formalizing the ECG protocol. Other languages like Glif, EON, Asbru, or GUIDE, [9] are more popular languages for the formal representation of guidelines in the medical domain. These languages can also detect anomalies in the medical protocols, but they are less formal than highly mathematical modelling languages like EVENT B, VDM and TLA$^+$. However, we expect that the informal nature of the other languages would reveal fewer anomalies during the formal development of the medical protocols. We plan to investigate other medical protocols using our refinement-based modelling approach. We have found that this

[4] http://www.guideline.gov/

approach is more applicable to rectifying the medical protocols as well as to obtaining the optimum solutions for diagnosis using the refinement-based incremental approach rather than any semi-formal techniques.

References

1. Lohr, K.N., Field, M.J.: Clinical practice guidelines: directions for a new program. In: Committee to Advise the Public Health Service on Clinical Practice Guidelines, United States and Institute of Medicine. National Academy Press, Washington, D.C. (1990)
2. Ten Teije, A., Marcos, M., Balser, M., van Croonenborg, J., Duelli, C., van Harmelen, F., Lucas, P., Miksch, S., Reif, W., Rosenbrand, K., Seyfang, A.: Improving medical protocols by formal methods. Artif. Intell. Med. 36(3), 193–209 (2006)
3. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press (2010)
4. Project RODIN: Rigorous open development environment for complex systems (2004), http://rodin-b-sharp.sourceforge.net/
5. Méry, D., Singh, N.K.: Technical Report on Interpretation of the Electrocardiogram (ECG) Signal using Formal Methods. Technical report, LORIA UMR 7503 (2011), http://hal.inria.fr/inria-00584177/en/
6. Shahar, Y., Miksch, S., Johnson, P.: The asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines. Artificial Intelligence in Medicine, 29–51 (1998)
7. Samson, M.M., Musen, M.A., Tu, S.W., Das, A.K., Shahar, Y.: Eon: A component-based approach to automation of protocol-directed therapy (1996)
8. Fox, J., Johns, N., Rahmanzadeh, A.: Disseminating medical knowledge: the proforma approach. Artificial Intelligence in Medicine 14(1-2), 157–182 (1998)
9. Peleg, M., Tu, S., Bury, J., Ciccarese, P., Fox, J., Greenes, R.A., Miksch, S., Quaglini, S., Seyfang, A., Shortliffe, E.H., Stefanelli, M., et al.: Comparing computer-interpretable guideline models: A case-study approach. JAMIA 10 (2003)
10. Wang, D., Peleg, M., Tu, S.W., Boxwala, A.A., Greenes, R.A., Patel, V.L., Shortliffe, E.H.: Representation primitives, process models and patient data in computer-interpretable clinical practice guidelines: A literature review of guideline representation models. International Journal of Medical Informatics 68(1-3), 59–70 (2002)
11. Isern, D., Moreno, A.: Computer-based execution of clinical guidelines: A review. International Journal of Medical Informatics 77(12), 787–808 (2008)
12. Miller, P.L.: A critiquing approach to expert computer advice: Attending. Pitman Publishing, Inc., Marshfield (1985)
13. Marcos, M., Berger, G., van Harmelen, F., ten Teije, A., Roomans, H., Miksch, S.: Using Critiquing for Improving Medical Protocols: Harder than It Seems. In: Quaglini, S., Barahona, P., Andreassen, S. (eds.) AIME 2001. LNCS (LNAI), vol. 2101, pp. 431–441. Springer, Heidelberg (2001)
14. Shiffman, R.N., Greenes, R.A.: Improving Clinical Guidelines with Logic and Decision-table Techniques: Application to Hepatitis Immunization Recommendations. Med. Decis. Making 14(3), 245–254 (1994)
15. Shiffman, R.N.: Representation of Clinical Practice Guidelines in Conventional and Augmented Decision Tables. Journal of AMIA 4(5), 382–393 (1997)
16. Miller, D.W., Frawley, S.J., Miller, P.L.: Using semantic constraints to help verify the completeness of a computer-based clinical guideline for childhood immunization. Computer Methods and Programs in Biomedicine 58(3), 267–280 (1999)

17. Bottrighi, A., Giordano, L., Molino, G., Montani, S., Terenziani, P., Torchio, M.: Adopting model checking techniques for clinical guidelines verification. Artif. Intell. Med. 48, 1–19 (2010)
18. Holzmann, G.J.: The model checker SPIN. IEEE Trans. Softw. Eng. 23, 279–295 (1997)
19. Clarke, E.M., Grumberg, O., Peled, D.: Model Checking. MIT Press (1999)
20. Prez, B., Porres, I.: Authoring and verification of clinical guidelines: A model driven approach. Journal of Biomedical Informatics 43(4), 520–536 (2010)
21. Rumbaugh, J., Jacobson, I., Booch, G. (eds.): The Unified Modeling Language reference manual. Addison-Wesley Longman Ltd., Essex (1999)
22. Warmer, J., Kleppe, A.: The Object Constraint Language: Getting Your Models Ready for MDA, 2nd edn. Addison-Wesley Longman Publishing Co., Inc., Boston (2003)
23. Schmitt, J., Hoffmann, A., Balser, M., Reif, W., Marcos, M.: Interactive Verification of Medical Guidelines. In: Misra, J., Nipkow, T., Karakostas, G. (eds.) FM 2006. LNCS, vol. 4085, pp. 32–47. Springer, Heidelberg (2006)
24. Bäumler, S., Balser, M., Dunets, A., Reif, W., Schmitt, J.: Verification of Medical Guidelines by Model Checking – A Case Study. In: Valmari, A. (ed.) SPIN 2006. LNCS, vol. 3925, pp. 219–233. Springer, Heidelberg (2006)
25. Balser, M., Coltell, O., Croonenborg, J.V., Duelli, C., Harmelen, F.V., Jovell, A., Lucas, P., Marcos, M., Miksch, S., Reif, W., Rosenbr, K., Seyfang, A., Teije, A.T.: Protocure: Supporting the development of medical protocols through formal methods. In: SCPG 2004. Studies in Health Technology and Informatics, vol. 101, pp. 103–108. IOS Press (2004)
26. Balser, M., Reif, W., Schellhorn, G., Stenzel, K.: Kiv 3.0 for Provably Correct Systems. In: Hutter, D., Traverso, P. (eds.) FM-Trends 1998. LNCS, vol. 1641, pp. 330–337. Springer, Heidelberg (1999)
27. Balzer, M., Duelli, C., Reif, W.: Formal Semantics of Asbru-An Overview. In: Integrated Design and Process Technology IPDT (2002)
28. Kosara, R., Miksch, S., Andreas, S.A., Votruba, P.: Tools for acquiring clinical guidelines in asbru (2002)
29. Seyfang, A., Miksch, S., Marcos, M., Wittenberg, J., Polo-Conde, C., Rosenbrand, K.: Bridging the gap between informal and formal guideline representations. In: 17th European Conference on Artificial Intelligence, Riva del Garda, Italy, pp. 447–451. IOS Press (2006)
30. Miksch, S., Hunter, J., Keravnou, E.T. (eds.): AIME 2005. LNCS (LNAI), vol. 3581. Springer, Heidelberg (2005)
31. Khan, M.G.: Rapid ECG Interpretation. Humana Press (2008)
32. Barold, S.S., Stroobandt, R.X., Sinnaeve, A.F.: Cardiac Pacemakers Step by Step. Futura Publishing (2004) ISBN 1-4051-1647-1
33. Bjorner, D.: Software Engineering 1-2-3. Texts in Theoretical Computer Science. An EATCS Series. Springer (2006)
34. Bjørner, D., Henson, M.C. (eds.): Logics of Specification Languages. EATCS Textbook in Computer Science. Springer (2007)
35. Chandy, K.M., Misra, J.: Parallel program design - a foundation. Addison-Wesley (1989)
36. Lamport, L.: Specifying Systems. The TLA+ Language and Tools for Hardware and Software Engineers. Addison-Wesley (2002)
37. Back, R.J., Xu, Q.: Refinement of fair action systems. Acta Inf. 35(2), 131–165 (1998)
38. Leavens, G.T., Abrial, J.R., Batory, D., Butler, M., Coglio, A., Fisler, K., Hehner, E., Jones, C., Miller, D., Peyton-Jones, S., Sitaraman, M., Smith, D.R., Stump, A.: Roadmap for enhanced languages and methods to aid verification. In: Fifth Intl. Conf. Generative Programming and Component Engineering (GPCE 2006), pp. 221–235. ACM (October 2006)
39. Atelier, B.: ClearSy Aix-en-Provence (F), Version 3.6 (2002)

Form Follows Function
Model-Driven Engineering for Clinical Trials

Jim Davies[1], Jeremy Gibbons[1],
Radu Calinescu[2], Charles Crichton[1], Steve Harris[1], and Andrew Tsui[1]

[1] Department of Computer Science, University of Oxford,
Wolfson Building, Parks Road, Oxford OX1 3QD, UK
http://www.cs.ox.ac.uk/firstname.lastname/
[2] Computer Science Research Group, Aston University,
Aston Triangle, Birmingham B4 7ET, UK
http://www-users.aston.ac.uk/~calinerc/

Abstract. We argue that, for certain constrained domains, elaborate model transformation technologies—implemented from scratch in general-purpose programming languages—are unnecessary for model-driven engineering; instead, lightweight configuration of commercial off-the-shelf productivity tools suffices. In particular, in the *CancerGrid* project, we have been developing model-driven techniques for the generation of software tools to support clinical trials. A domain metamodel captures the community's best practice in trial design. A scientist authors a trial protocol, modelling their trial by instantiating the metamodel; customized software artifacts to support trial execution are generated automatically from the scientist's model. The metamodel is expressed as an XML Schema, in such a way that it can be *instantiated by completing a form* to generate a conformant XML document. The same process works at a second level for trial execution: among the artifacts generated from the protocol are models of the data to be collected, and the clinician conducting the trial instantiates such models in reporting observations—again by completing a form to create a conformant XML document, representing the data gathered during that observation. Simple standard form management tools are all that is needed. Our approach is applicable to a wide variety of information-modelling domains: not just clinical trials, but also electronic public sector computing, customer relationship management, document workflow, and so on.

It is the pervading law of all things organic and inorganic,
Of all things physical and metaphysical,
Of all things human and all things super-human,
Of all true manifestations of the head,
Of the heart, of the soul,
That the life is recognizable in its expression,
That form ever follows function. This is the law.

Louis Sullivan [33]

1 Introduction

Randomized controlled trials are considered to be the 'gold standard' for experiments in medicine. They provide the most reliable evidence supporting or refuting a scientific hypothesis, such as that 'treatment X cures more patients suffering from disease D than does treatment Y'. An experiment is designed: treatment regimes X and Y will be specified; patients suffering from disease D will be recruited; recruits will be stratified into groups with similar relevant characteristics, based on factors such as age, gender and lifestyle; patients within each group will be allocated at random to treatment X or treatment Y; the results will be analyzed to determine whether or not the difference in effect of the treatments is statistically significant. So far, so good.

From a software point of view, a clinical trial is largely an exercise in data management: observations have to be specified, collected, recorded, integrated, and analyzed. But the software engineering aspects of setting up and running a clinical trial are not trivial. Two particular problems that we will address involve *data integration* and *tool generation*.

The data integration problem occurs because medical researchers want to be able to combine the results of multiple trials, a process known as *meta-analysis*. It is often the case that a single trial in isolation does not have adequate statistical power to yield a robustly significant conclusion. Nevertheless, if sufficiently many trials have been conducted, investigating sufficiently similar hypotheses and collecting sufficiently similar data, it may be possible to pool the results to greater effect: *"...the drug Tamoxifen—an oestrogen blocker that may prevent breast cancer cells growing—was the object of forty-two studies world-wide, of which only four or five had shown significant benefits. But this did not mean that Tamoxifen did not protect against breast cancer. When we put all the studies together it was blindingly obvious that it does..." [34].* In other situations, the meta-analysis aims at evaluating new hypotheses that are formulated long after the completion of the trials that originally collected the data involved—in this case, data from trials investigating quite different hypotheses may be integrated.

Either way, for meta-analysis to be possible, it is necessary to capture and curate metadata expressing the 'semantics' of the data; only then is it possible to determine whether data collected in different trials are commensurate, and if so, how to relate them. For example, when measuring blood pressure, it is not enough to record a pair of numbers, or a pair of pressures, or a pair of measurements in mmHg, or even to indicate that these represent systolic and diastolic pressure; it is also necessary to know how that data was collected (at rest, or after five minutes on a treadmill?), and maybe factors such as who collected it (in the clinic by a professional, or at home by the patient?) and how recent and reliable it is. This semantic metadata is an essential part of the context of the data, and logically forms part of the model of the trial—alongside more syntactic metadata such as the name of the trial and the types of data items.

As for tool development, standard practice in clinical trials management these days is to pass a textual document containing the trial protocol over to database programmers in the clinical trials unit, or to consultants from a trials management software provider, who will use it as guidance in manually configuring an information management system for this particular trial. This practice causes numerous problems. Firstly, it induces delays: it is usually the case that some to-ing and froing is needed between the database programmers and the medical researchers to get the details right, but the medics are too busy to respond immediately, and the start of the trial is often delayed because the software is not ready. Secondly, it is costly. This is not such a problem for a big 'phase III' trial for measuring effectiveness of a treatment: this will have thousands of participants and a stable design, so the software development will form only a small proportion of the overall cost, and anyway such a trial is likely to be commercially funded rather than run on a shoestring. But it certainly becomes a problem for 'early-phase' exploratory studies, which are much smaller, more dynamic, and poorly funded. Thirdly, it is not uncommon for an early-phase trial protocol to undergo changes during the execution of the trial, requiring adjustments to software components of the associated trial management system; current practice is to implement these changes through manually modifying the underlying code, running the risk of introducing software bugs when the system is in production use. And finally, bespoke database design on a per-trial basis is unlikely to promote the consistency and interoperability needed for meta-analysis.

All four of these generation issues could be addressed if the development of the software tools needed to support trial execution could be automated. Fortunately, there is essentially enough information in the trial protocol—which needs to be written anyway, not least for the purposes of regulatory approval—to completely determine the relevant software artifacts, either from scratch or by configuring more generic components. If the protocol were written in a more structured format—that is, as a formal model, rather than merely a textual description, of the trial—then both the prose and the code could generated from it by suitable processing, and any adjustments required because of changes to the trial protocol can be made without risky manual intervention at the level of code. Moreover, as we have seen, the annotation of the data descriptions in the trial model with semantic metadata will make that model doubly useful, as a basis for supporting meta-analysis in addition to being a specification for a software system.

In other words, clinical trials management is crying out for a model-driven engineering (MDE) approach.

Contribution

Our main contribution in this paper is a demonstration that model-driven engineering need not involve the modeller—who is ideally a domain expert rather than a developer—in 'programming' in a textual or graphical notation; rather, in certain constrained domains, the modelling task is sufficiently formulaic that it can be conducted simply by completing forms. Our argument is illustrated by way of examples drawn from clinical trials management; but we also discuss its applicability to other information domains, such as electronic government. In this paper we focus on the model-driven aspects of the problem; data integration is discussed in a companion paper [12].

2 The CancerGrid Project

The CancerGrid project [7] was instigated to address the twin problems of interoperability and generativity in clinical trials, taking a model-driven approach to the development of trials management tools. It was funded in the first instance for three years from 2005 by the UK Medical Research Council, with the involvement of five UK universities: Cambridge (specializing in oncology), Oxford (software engineering), University College London (semantic modelling), Birmingham (clinical trials management), and Belfast (telemedicine). Oxford University Software Engineering Programme and the Cancer Research UK Cambridge Research Institute have been continuing the work.

An early activity of the project was the reification of the *Consolidated Standards of Reporting Trials* (CONSORT) statement [27] as a domain model. The CONSORT statement is a widely-adopted checklist of thirty-two items, intended to capture best practice in reporting clinical trials: name, date, sponsor, unique unique trial number, target recruitment size, randomization protocol and stratification variables, eligibility criteria, case report form structure, clinical interventions, and so on. We expressed this checklist as a class model of well-designed trials; this model is described in detail in a technical report [19], and the principles behind the model discussed in a companion paper [12].

An intriguing aspect of the CancerGrid approach is the parallels it reveals between the 'design-time' and 'run-time' stages in the lifecycle of a trial. In the first stage, a scientist is designing a trial protocol, following the guidelines set out in the CONSORT statement; in the second stage, a clinician is conducting a trial, following the guidelines set out in the trial protocol. Both cases involve instantiating a schema: the trial protocol instantiates the CONSORT class model, but it determines schemas for data collection and reporting at significant events during the trial, which are themselves instantiated by the clinician when entering data. We therefore call our model of the CONSORT statement the *metamodel*, because each protocol that instantiates it is itself a model of a particular trial. This parallel is discussed in more detail in Section 4, where the workflow of trial design and execution is presented.

Much of the interoperability requirement pivots on some kind of consensus on—or at least, machine-processable documentation of—the *common data elements* (CDEs) being recorded. There can be no magic here: if two trials have collected incompatible data, or one of them has provided insufficient metadata to determine compatibility, then their results cannot usefully be integrated. On the other hand, it is very difficult to arrange for prior universal agreement on compatible data elements across a large, heterogeneous, and long-lived community.

The approach to this dilemma that we have taken on the CancerGrid project is realist rather than idealist. We have developed tools to support communities in deciding on, recording, and disseminating data standards; but there is no need for all parties to commit to using the same standard. We have developed the *CancerGrid Metadata Registry* (cgMDR) [8], an open-source implementation of the ISO 11179 metadata registry standard [20]. This is robust enough for widespread use—it is currently being adopted by the US National Cancer

Institute (http://www.cagrid.org/display/MDR/), for example—while still being lightweight enough for individual trials units to install their own copy to support local practice and customization. Data elements (for example, 'blood pressure on induction into study, measured at rest') are curated in the metadata registry, and referenced in the trial protocol; the metadata reference is preserved in the software artifacts generated from the protocol—data entry forms, database schemas, spreadsheets, and so on—ensuring that all the data maintain their semantic annotations throughout their journey through the system.

As the name suggests, the original plan in 2005 was to use the then-emerging 'grid' technologies as a basis for the implementation. This turned out to be impractical: the toolkits were relatively unsophisticated, requiring considerable programming effort to duplicate the functionality of applications that were already available to our target users; moreover, 'grid computing' in the sense of large-scale computational or storage clusters is not relevant to this particular problem. The development activity was thus refocussed upon the production of software that worked to extend and configure applications that are widely used and available within the UK National Health Service (and, indeed, throughout government and industry): specifically, Microsoft Office and SharePoint Server. The project's focus upon the requirements of clinical researchers, and the recognition that these requirements can be met partly through the (automated) enhancement and configuration of office productivity applications, has led to changes in attitudes. As one team member put it: "We used to be hung up on open source; now we're really focussed on open standards."

Also as suggested by the name, the initial focus was on cancer clinical trials; the first validations were on three breast cancer trials [28,29,16]. The tools have also been demonstrated in an Anglo-Canadian cancer clinical study, in which clinical data and tissue sample metadata from five centres were integrated automatically on the basis of declared semantics in cgMDR, allowing an analysis across 4000 samples that would have been impractical under existing approaches; and in a proof-of-concept usage on the US Veterans' Health Administration Cooperative Studies Program, where forms have been generated for serious adverse event reporting in a rheumatoid arthritis study that allow the incorporation of metadata directly from the US National Cancer Institute Thesaurus.

Current applications of the CancerGrid approach include: *Accelerating Cancer Research Using Semantics-Driven Technology* [3], funded by Microsoft Research, exploring the extension from phase III to early-phase studies; *Evolving Health Informatics* [14], funded by Research Councils UK, working with colleagues in the Centre for Clinical Vaccinology and Tropical Medicine at the University of Oxford to demonstrate applicability to infectious disease control; *Hospital of the Future*, aiming to improve patient outcomes through information-driven management; the *Data Support Service*, funded by the UK Medical Research Council (MRC), to retrospectively catalogue the data collected in some of the MRC's valuable long-running studies; and the *Union of Light-Ion Centres in Europe*, funded by the European Union Seventh Framework Programme, to curate experimental results in particle therapy.

3 Forms-Based Model-Driven Engineering

The Object Management Group's *Model-Driven Architecture* [26] aims to raise the level of abstraction in software development, by separating higher-level specifications of system functionality from lower-level descriptions of the implementation of that functionality on a particular platform. *Platform-independent models* (PIMs) abstract away from technical details of implementation, such as representation in terms of CORBA objects or SOAP-based web services; these technical details only appear in *platform-specific models* (PSMs). The transformations from PIMs to PSMs (and for that matter, between PIMs and between PSMs) might be fully automated, or might require some degree of human intervention.

Models must be expressed in some structured notation, such as UML or XML. The construction of models has a number of benefits over going straight to code. Firstly, a higher-level PIM is, presumably, simpler than a lower-level PSM, being less cluttered with details; this makes it easier to analyse, to test, and to reason about. Secondly, the effort involved in constructing and refining PIMs may be amortized over multiple PSMs, encouraging investment in reusable assets. And thirdly, interoperability between systems can be more clearly specified in terms of PIMs, with integration mechanisms consequently inferred from PSMs. If all one has is code, then all one can do is to execute it; whereas higher-level models may be put to multiple uses.

The approach to model-driven engineering taken in the CancerGrid project is a document-centric one: *data models as document schemas*; *entities as conformant documents*; *authoring as form completion*; and *model transformations as schema mappings*. Moreover, there is a pleasing symmetry between design-time and run-time activities: the former consists of constructing a system model as a document that conforms to a domain metamodel, the latter consists of recording an event as a document that conforms to a data model, and both involve instantiation via form completion. This view is perhaps not a universally appropriate approach to model-driven engineering—more complex modelling notations might not be readily viewed as document formats, and form-filling might not be the most convenient way of authoring models in such notations—but it works in restricted domains, and in particular in clinical trial design.

In a way, we are heading in the opposite direction to that taken by the *Executable UML* camp [23]. Rather than thinking of models as programs written in a high-level programming language, we think of them as plain data; the programs are simple generic interpreters of this data, written once and reused many times. For example, the trial metamodel is manifested as an XML Schema, and the 'trial designer' application is simply an off-the-shelf tool (in our case, Microsoft InfoPath) that allows a scientist to design a trial by completing a form derived from that schema. In this sense, we are following Brooks' advice [4]: "*Show me your flowchart and conceal your tables, and I shall continue to be mystified. Show me your tables, and I won't usually need your flowchart; it'll be obvious*", or, as later pithily paraphrased by Steele [31]: "*Smart data structures and dumb code works a lot better than the other way around.*"

4 Implementation Approaches

The general principles of data models as document schemas, entities as conformant documents, authoring as form completion, and model transformations as schema mappings can be realized in a variety of ways. Our current implementation uses an off-the-shelf product for form-filling—namely Microsoft InfoPath [24], an application that supports the design and completion of XML-based data entry forms, forming part of the Microsoft Office productivity suite. This has turned out to be the approach most attractive to our target audience of medical researchers and clinicians, because they are all familiar with the Microsoft Office interface, and they all have the software pre-installed on their desktop computers. This approach is described in Section 4.1. However, the general principles are in no way tied to this particular realization, and in Section 4.2 we briefly discuss an alternative implementation approach constructed from first principles.

4.1 Off-the-Shelf Implementation

The workflow entailed by trial design and execution is illustrated in Figure 1. The steps involved are expanded below, and illustrated by means of a paediatric vaccinology study in Kathmandu [13] that we are currently supporting.

First, representatives of the data community design a metamodel of the kinds of experiment in which they are interested, capturing the design in the form of an XML schema; this may be done once and for all, or more realistically, it is subject to relatively infrequent updates. In our case, the general discussion had already taken place in the medical research community, resulting in the existing CONSORT statement [27], which has remained stable for over a decade. We expressed the CONSORT metamodel as a UML class model, and exported the metamodel as an XML schema. This metamodel is quite elaborate, but is partitioned into packages defining trial description, patient eligibility, randomization, treatments,

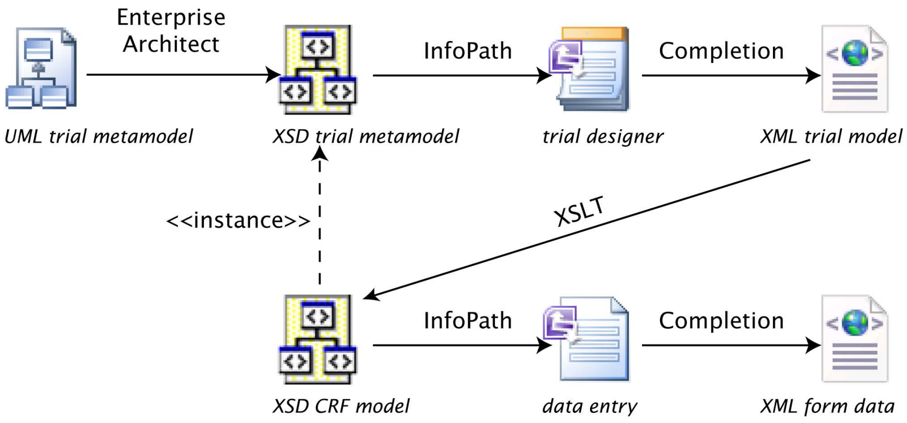


Fig. 1. The forms-based workflow of clinical trial design and execution; icons denote artifacts, and solid arrows denote transformations between artifacts

case report forms and form controls, and security policy; a fragment representing form controls is shown in Figure 2. This aspect of the CancerGrid philosophy is described in more detail elsewhere [12]. (In addition, but also once and for all, we developed a number of trial-independent software artifacts, such as generic randomization and case report form management web services. And of course, we are exploiting off-the-shelf applications such as InfoPath and SharePoint.)

Second, a medical researcher planning a clinical trial designs the trial protocol. The *trial designer* application that they use to do so is just InfoPath configured with the trial metamodel. Designing a trial amounts to completing the forms that InfoPath presents; the result is an XML document modelling this particular trial that, by construction, conforms to the schema from the first step—and hence

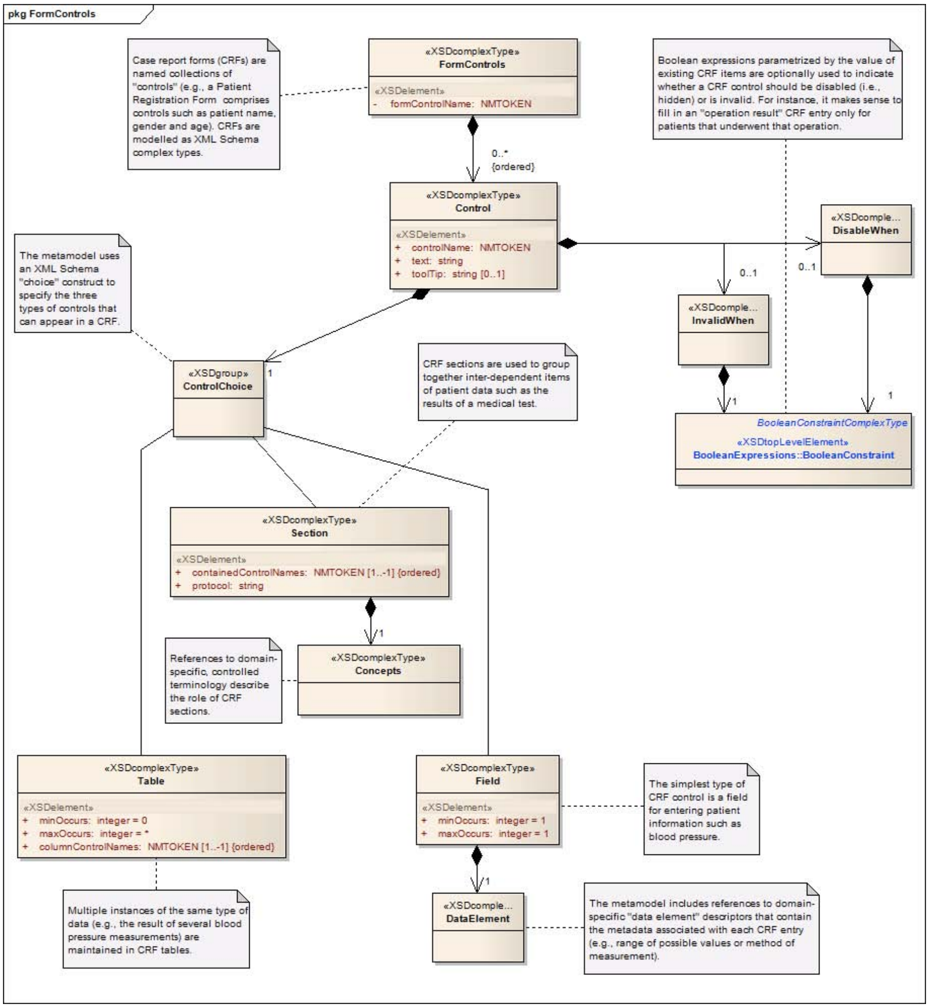
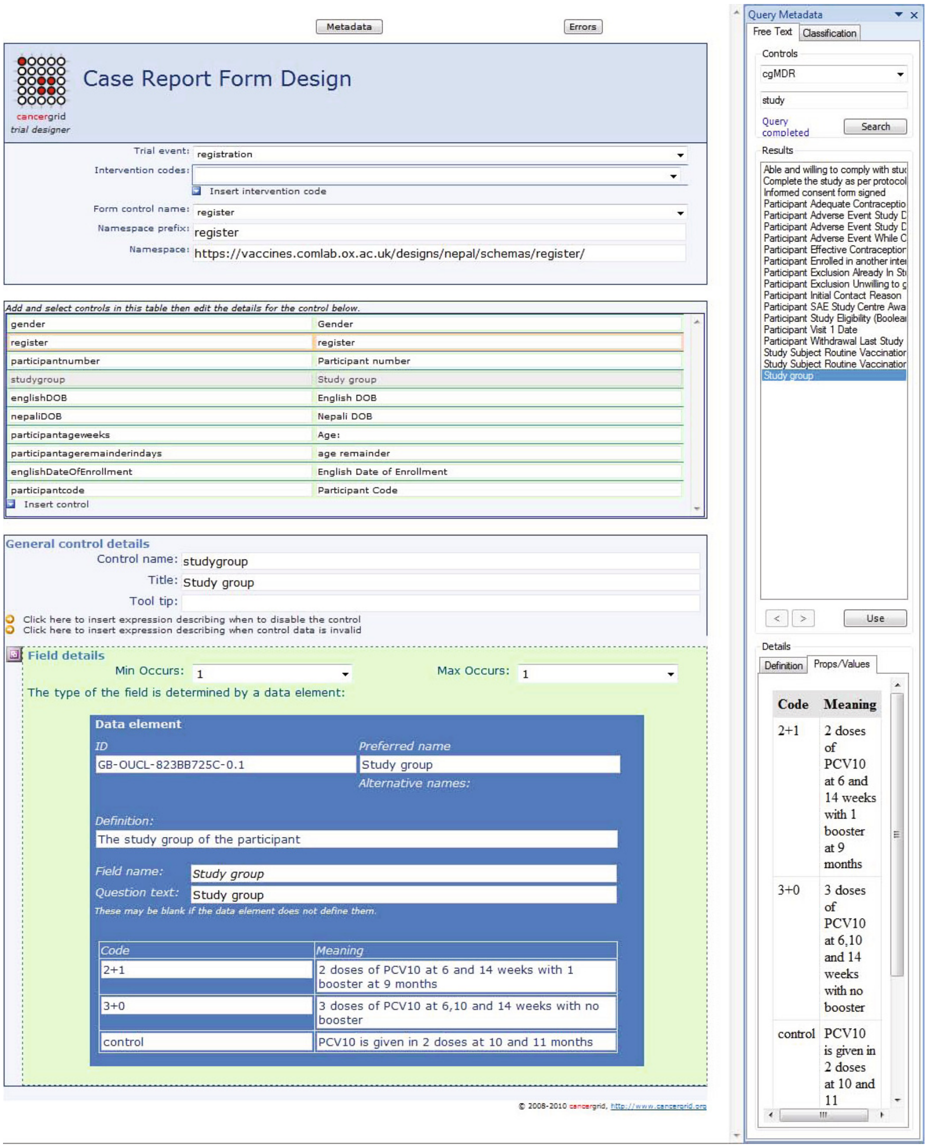

pkg FormControls
Case report forms (CRFs) are named collections of "controls" (e.g., a Patient Registration Form comprises controls such as patient name, gender and age). CRFs are modelled as XML Schema complex types.
«XSDcomplexType» FormControls
«XSDelement» - formControlName: NMTOKEN
0..* {ordered}
Boolean expressions parametrized by the value of existing CRF items are optionally used to indicate whether a CRF control should be disabled (i.e., hidden) or is invalid. For instance, it makes sense to fill in an "operation result" CRF entry only for patients that underwent that operation.
«XSDcomplexType» Control
«XSDelement» + controlName: NMTOKEN + text: string + toolTip: string [0..1]
«XSDcomple... DisableWhen
0..1
0..1
«XSDcomple... InvalidWhen
1
1
The metamodel uses an XML Schema "choice" construct to specify the three types of controls that can appear in a CRF.
«XSDgroup» ControlChoice
1
CRF sections are used to group together inter-dependent items of patient data such as the results of a medical test.
BooleanConstraintComplexType
«XSDtopLevelElement» BooleanExpressions::BooleanConstraint
«XSDcomplexType» Section
«XSDelement» + containedControlNames: NMTOKEN [1..-1] {ordered} + protocol: string
1
References to domain-specific, controlled terminology describe the role of CRF sections.
«XSDcomplexType» Concepts
«XSDcomplexType» Table
«XSDelement» + minOccurs: integer = 0 + maxOccurs: integer = * + columnControlNames: NMTOKEN [1..-1] {ordered}
«XSDcomplexType» Field
«XSDelement» + minOccurs: integer = 1 + maxOccurs: integer = 1
The simplest type of CRF control is a field for entering patient information such as blood pressure.
1
Multiple instances of the same type of data (e.g., the result of several blood pressure measurements) are maintained in CRF tables.
«XSDcomple... DataElement
The metamodel includes references to domain-specific "data element" descriptors that contain the metadata associated with each CRF entry (e.g., range of possible values or method of measurement).

Fig. 2. UML class model of the Form Control package of the CancerGrid metamodel

also to the CONSORT statement. (In contrast, CONSORT conformance is not guaranteed with the current practice of writing the protocol in plain English.) Figure 3 shows a screenshot of InfoPath being used to design a case report form for recording study participant registration; the highlighted element is the 'study group' (a three-way enumeration) into which the participant is placed, and a version of the underlying XML representation of this piece of the form, heavily edited for space and readability, is shown in Figure 4(a).



Case Report Form Design

Trial event:	registration
Intervention codes:	
	Insert intervention code
Form control name:	register
Namespace prefix:	register
Namespace:	https://vaccines.comlab.ox.ac.uk/designs/nepal/schemas/register/

Add and select controls in this table then edit the details for the control below.

gender	Gender
register	register
participantnumber	Participant number
studygroup	Study group
englishDOB	English DOB
nepaliDOB	Nepali DOB
participantageweeks	Age:
participantageremainderindays	age remainder
englishDateOfEnrollment	English Date of Enrollment
participantcode	Participant Code

Insert control

General control details

Control name:	studygroup
Title:	Study group
Tool tip:	

Click here to insert expression describing when to disable the control
Click here to insert expression describing when control data is invalid

Field details

Min Occurs: 1 Max Occurs: 1

The type of the field is determined by a data element:

Data element

ID	Preferred name
GB-OUCL-823BB725C-0.1	Study group
	Alternative names:

Definition: The study group of the participant

Field name: Study group
Question text: Study group
These may be blank if the data element does not define them.

Code	Meaning
2+1	2 doses of PCV10 at 6 and 14 weeks with 1 booster at 9 months
3+0	3 doses of PCV10 at 6,10 and 14 weeks with no booster
control	PCV10 is given in 2 doses at 10 and 11 months

© 2008-2010 cancergrid, http://www.cancergrid.org

Query Metadata

Free Text | Classification

Controls
cgMDR
study
Query completed
Search

Results
Able and willing to comply with stu
Complete the study as per protocol
Informed consent form signed
Participant Adequate Contraceptio
Participant Adverse Event Study D
Participant Adverse Event Study D
Participant Adverse Event While C
Participant Effective Contraceptior
Participant Enrolled in another inte
Participant Exclusion Already In St
Participant Exclusion Unwilling to g
Participant Initial Contact Reason
Participant SAE Study Centre Awa
Participant Study Eligibility (Boolea
Participant Visit 1 Date
Participant Withdrawal Last Study
Study Subject Routine Vaccinatior
Study Subject Routine Vaccinatior
Study group

Use

Details
Definition | Props/Values

Code	Meaning
2+1	2 doses of PCV10 at 6 and 14 weeks with 1 booster at 9 months
3+0	3 doses of PCV10 at 6,10 and 14 weeks with no booster
control	PCV10 is given in 2 doses at 10 and 11

Fig. 3. Using InfoPath to design a case report form

Third, the XML document recording the trial protocol determines numerous software artifacts relating to the trial: clinical interventions, datasets and collection procedures, software configurations for services such as randomization and validation, documentation, and so on. Each trial-specific artifact essentially instantiates the template for such artifacts included in the metamodel—hence the dashed realization arrow in Figure 1 stereotyped '⟪instance⟫', in a slight abuse of notation. (To avoid clutter, the only trial-specific artifact shown in the figure is the model of a case report form (CRF). Other artifacts are also models, but may be models of entities such as services, documents, or workflows, rather than of forms.) The specification of each artifact is obtained by traversing the XML document, extracting the corresponding parts, and transforming them into the appropriate format: XML Schema, XSL Formatting Objects, WSDL, and so on. Traversal, extraction, and transformation is specified as a collection of XSLT stylesheets, written once only, for all trials based on the same version of the metamodel. In particular, the data to be collected by the clinician conducting the trial—and hence the structure of the form on which this data is recorded—is specified by an XML schema. The data manager in the trials unit generates all these artifacts automatically, and deploys them in the unit's web portal for access by clinicians. Continuing the example, the XML defining the participant study group is used to generate an XML Schema for the data that should be collected; the corresponding portion of that schema, again heavily edited for readability, is shown in Figure 4(b).

Finally, the clinician in the unit running the trial conducts a consultation with a participant in the trial, makes some clinical observations, and needs to record the data so obtained. The *data entry* application that they use to do so is just InfoPath configured with the model of the relevant data. Data entry amounts to completing the form that InfoPath presents; the result is an XML document recording this event that, by construction, conforms to the appropriate schema from the third step, and which may be stored in an XML database for subsequent analysis and study. In the example, choices for the participant study group are presented as a dropdown list; selection from this list results in the creation of the XML in Figure 4(c).

Notice especially that the execution-time actions of the clinician filing a case report closely mirror the design-time actions of the scientist modelling the trial: both actions are manifested as completion of a form to generate a document that conforms to a certain schema. The actors may play different roles in the overall process, but both are domain specialists rather than software professionals, and for both participants a familiar generic application configured with an appropriate model is a suitable tool.

Notice also a significant benefit of the model-driven approach: the model can be used for other purposes than just 'execution'. For example, as well as the software artifacts derived from a trial protocol, a textual description has to be generated for submission to the relevant regulatory bodies. We have implemented a simple 'reporting tool' to generate such a description.

(a)
```
<controlName>studygroup</controlName>
<text>Study group</text>
<minOccurs>1</minOccurs>
<maxOccurs>1</maxOccurs>
<data-element>
  <id>GB-OUCL-823BB725C-0.1</id>
  <definition>The study group of the participant</definition>
  <valid-value>
    <code>2+1</code>
    <meaning>2 doses of PCV10 at 6 and 14 weeks with 1 booster at 9 months</meaning>
  </valid-value>
  <valid-value> ... </valid-value>
</data-element>
```

(b)
```
<xs:element name="studygroup" type="register:SimpleType_studygroup"/>
<xs:simpleType name="SimpleType_studygroup"
               sawsdl:modelReference="GB-OUCL-823BB725C-0.1">
  <xs:annotation><xs:documentation source="definition">
    The study group of the participant
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="2+1" />
    <xs:enumeration value="3+0" />
    <xs:enumeration value="control" />
  </xs:restriction>
</xs:simpleType>
```

(c)
```
<register:studygroup>3+0</register:studygroup>
```

Fig. 4. Fragments of XML representing (a) a trial model, (b) a data schema derived from the model, and (c) recorded results conforming to the schema

4.2 Implementation from First Principles

The implementation approach described in Section 4.1 is straightforward and robust (on account of being a simple specialization of a proven off-the-shelf framework), and is attractive to our particular end users (on account of their familiarity with this framework). However, a closed commercial off-the-shelf framework may not be appropriate for all environments; so as an exercise in risk mitigation, we also developed a proof-of-concept implementation from first principles. We present a brief outline here; space limitations preclude a more detailed description.

The thoroughness with which the CONSORT statement specifies the requirements for randomized clinical trials enabled us to develop, once and for all, trial-independent software artifacts for managing clinical trial data. We used generic programming techniques to implement these artifacts as .NET web services parametrized by form models. This specialization of the generic trial services using trial-specific form datatypes is analogous to the configuration of Microsoft InfoPath using form schemas.

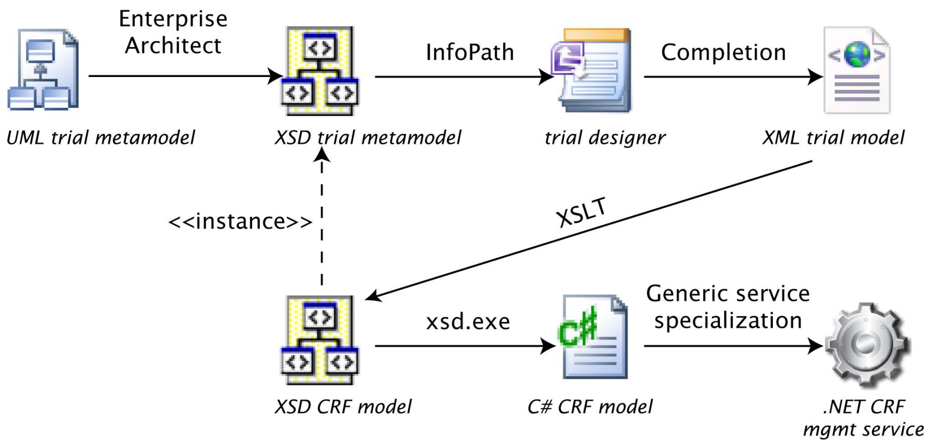


Fig. 5. The forms-based workflow of clinical trial service generation; as in Figure 1, icons denote artifacts, and solid arrows denote intermediate transformations

Figure 5 illustrates the workflow involved in generating CRF management services for a clinical trial. The first part—leading to the generation of an XML schema model of the case report form—is identical to the first part of the workflow presented in Section 4.1 (see Figure 1). For the second part of the service generation workflow, the off-the-shelf XML Schema Definition Tool (Xsd.exe) [25] is used to produce a C# class corresponding to the XML schema model of the case report form—in other words, a C# model of the form. This C# CRF model is then used to specialize the generic CRF management service, and thus to obtain the trial-specific CRF management service.

The same process is entailed by the generation of other similar run-time artifacts, including services for the management of patients, staff (e.g., clinicians, research nurses and statisticians) and locations (i.e., the healthcare research units involved in the clinical trial). Indeed, the same process could be used at design time as well, for authoring the trial protocol: this could use a generic data entry form completion application, analogous to InfoPath but parametrized by a C# class rather than by an XSL schema.

5 Evaluation

The approach to model-driven engineering of clinical trials management software that we have described—representing data models as document schemas, software artifacts as conformant documents, authoring as form completion, and model transformations as schema mappings—has been validated in a number of experiments, as outlined in Section 2. These have included both highly regular phase III trials and more exploratory early-phase studies; both metadata-only and metadata-plus-data exercises; both prospective data capture and retrospective data cataloguing; and both national and international communities.

The results are promising, although still on a small scale; we have in gestation some more ambitious experiments for larger-scale validation.

We have been able to demonstrate, for a wide variety of types of clinical study, that bespoke applications, tailored to specific study designs, can be delivered quickly through the automatic configuration of (relatively) low-cost general-purpose software, and that the approach can be validated for the purposes of regulatory submissions. Anecdotal evidence suggests that the reduction in development time may range between 20% and 80%, depending upon the organizational context. Based on the early evaluations mentioned in Section 2, it seems that the reduction in licensing costs can be measured in terms of the cost of a Microsoft Office SharePoint Server select license (of the order of £1000, covering arbitrarily many studies) against that for a typical solution based on commercial domain-specific tools (which “would cost in the range of hundreds of thousands of dollars” [17]).

The major benefit of our forms-based model-driven approach is that it enables the domain expert (i.e., the scientist modelling the trial) to take part in the development of the software, reducing the errors and delays that often occur when informal models are interpreted by an IT expert who lacks domain knowledge. Additionally, both the effort to bring a live software system back into line with a new version of a trial protocol, and the risk of introducing defects in doing so, are eliminated almost completely by automating the process. Another (still unexplored, but nevertheless considered) benefit is the ability to use software instances generated from preliminary protocol versions for testing and iteratively improving the protocol, with minimal development effort; in contrast, current practice delays the identification of flaws in the trial protocol until very late in the development cycle.

6 Conclusions

We have described an approach to model-driven engineering for information-rich domains that can be characterized by a domain metamodel. Model-building is a matter of instantiating the metamodel; when the metamodel is sufficiently regular, that process can reduce to simple form-filling, and can be conducted by domain experts with no programming skills. For many tasks, the same process is applicable a second time when running the application: the model determines a data format, and data gathering again reduces to form-filling, constructing data items that conform to the model. What is most interesting is that modern productivity frameworks—such as the SharePoint and InfoPath components of Microsoft Office—are sufficiently powerful and flexible to support this kind of application, almost without having to resort to low-level implementation in a general-purpose programming language. This is evidence of a raise in the level of abstraction provided by current mainstream software tools.

It has been gratifying to see that the ideas we have developed are much more widely applicable than originally envisaged. Of course, we expected that software engineering efforts related to breast cancer ought to be readily translatable to other types of cancer, and we hoped that its area of applicability would also embrace other diseases; our results with communities working on rheumatoid arthritis and in vaccinology have vindicated that hope.

But we have been pleasantly surprised to learn that essentially the same approach is also highly relevant in the field of electronic government. This too turns out to be largely a problem of data integration: the former UK Prime Minister Tony Blair coined the term 'joined-up government' as a vision for how different government departments ought to—but generally do not at present—interact [1]. Moreover, electronic governance is also a domain in which model-driven generation of software artifacts would be extremely helpful: accountability of public servants requires government information systems to be transparent, and the monopoly typically held by the incumbent government requires the systems to be trustworthy. Our ideas in this area are still under development, but we have some preliminary results [10,15,11], and we are discussing further progress with the UK Public Sector Object Model group and with the Scottish Government.

Broadly speaking, we expect the approach to be applicable to any semantically rich domain in which there is: a relatively stable metamodel of the domain; a 'design phase' consisting of instantiating the metamodel to yield a model of a particular instance, which can be used to configure generic software tools; and an 'execution phase' in which entities conforming to the model are created. For example, one could use the approach for a generic conference management system. The basis is a metamodel of academic conferences. The conference chair 'designs' the particular conference by instantiating the metamodel, specifying properties such as whether there is an author response period, whether reviews are double-blinded, how many reviewers each paper should have, and so on. 'Execution' consists of creating entities such as 'submissions' and 'reviews' that conform to conference-specific aspects of the model. The reader can doubtless think of many similar configurable information-gathering exercises.

One aspect of ongoing work is to extend the scope of the metamodel to cover also the temporal aspects of a clinical trial. Although the trial protocol provides structured specifications of static aspects, in terms of common data elements, the dynamic aspects—when interventions should occur—are described only in free text (see the 'meaning' fields in Figure 3). These too could be specified in a structured format in the protocol, in a workflow modelling notation such as BPEL or BPMN, and then used to generate scheduling tools for trial execution. We have conducted some preliminary studies on using such workflow notations to specify and check trial safety properties such as drug interactions [38,39], but have not yet integrated this work with the rest of the CancerGrid toolchain. The biggest challenge will be to allow the trial designer to describe the temporal aspects of the trial in sufficient detail, without degenerating into a full-blown programming exercise; we hope that *workflow patterns* [37] and *property specification patterns* [40] will be helpful in this regard.

7 Related Work

The US cancer Biomedical Informatics Grid (caBIG) [36] have a metamodel of clinical trials [21], and have the sharing of cancer clinical data as a primary objective. However, their caCORE software development kit [35] handles only the generation of web service stubs for some aspects of cancer informatics, requiring manual intervention by software developers in order to fill out the logic. Their trial models, in contrast to ours, are not detailed enough to specify the behaviour of these services, and so their generation process can only go as far as an outline rather than a complete implementation.

There are many data interchange formats for medical data, especially for healthcare records—for example, the EN 13606 standard for electronic health record communication (www.en13606.org) and its reference implementation (www.openehr.org), the Health Level Seven messaging standards (www.hl7.org), and the CDISC Operational Data Model (www.cdisc.org/odm). Any of them could be used in conjunction with the CancerGrid metamodel-based tools; but they are more descriptive than prescriptive—none of them are designed to support the scientist in specifying *which* data elements to collect, and when.

Other medical informatics projects are concerned with the integration of data from multiple, distributed databases. For example, VOTES [32] integrates distributed medical data pertaining to a single patient, so that candidate patients for new clinical trials can be identified easily; the query forms used by the VOTES portal resemble those discussed here, but they are encoded manually by software developers familiar with the internal structure of the data sources, rather than being generated from a higher-level model. The PRATA system [9] addresses the XML integration of data extracted from multiple, distributed databases, but based on a user-specified XML schema that requires inside knowledge of the data sources.

Sierra *et al.* [30] describe a document-centric approach to application development, similar to ours in the sense of involving a domain metamodel, models as structured documents, and model-driven generation of software artifacts; but they expect domain experts to author models in markup languages like XML rather than by simpler form completion. McLaren and Wicks [22] discuss a generative framework using XML; they present an 'indirect' approach using XSLT to turn an XML model into an XML—or other format of—software artifact, and a 'direct' approach using traditional (for example, Java) code to parse and interpret XML on the fly. The indirect approach is more agile, and is better while requirements are evolving rapidly; but with a large system, the transformation process may take considerable time, and McLaren and Wicks argue that the 'faster direct implementation is necessary once requirements are well defined'. Our approach is at the indirect end of the spectrum, with very little hand-crafted code; even with our two-stage process, in which end users rather than software engineers initiate model transformations, we do not find the transformation process too time-consuming: maybe the technological landscape has changed enough since 2001 for processing speed no longer to be quite such a concern.

Finally, it is interesting to view our forms-based approach from the perspective of domain-specific languages [18]: one might say that forms present a third style of DSL, in addition to—perhaps in between—the familiar textual and graphical styles.

Acknowledgements. We would like to acknowledge the other members of the CancerGrid team and related projects—James Brenton, Carlos Caldas, Marta Kwiatkowska, Peter Maccallum, Sylvia Nagl, Aadya Shukla, Matthew Snape, Tianyi Zang—for their contributions towards the ideas presented here. We are also very grateful to the Medical Research Council (grant number G0300648), Research Councils UK (grant number EP/F059345/1), the Engineering and Physical Sciences Research Council (grant number EP/H019944/1), the EU Framework Programme 7 (grant number 228436), and Microsoft Research, for funding this work.

Some aspects of the CancerGrid project have also been published elsewhere. An early paper [2] set out the original vision, involving semantic web technology, computational grids, and computer-supported collaborative working; as discussed in this paper, many of these original ideas turned out to be unworkable in practice, and the project soon changed direction. A pair of related papers [5,6] present formal specifications of an early prototype of our form-based approach, but programmed from scratch in Java rather than by configuring an off-the-shelf application (analogous to the first-principles implementation discussed in Section 4.2). A companion paper [12] focusses on the metadata aspects supporting shared semantics, rather than on the model-driven technology as described here. An extended abstract [13] briefly discusses the Kathmandu vaccinology study used as a running example in Section 4.

References

1. Blair, T.: Modernising government. UK Cabinet Office white paper CM 4310, UK Government (March 1999), `http://www.archive.official-documents.co.uk/document/cm43/4310/4310.htm`
2. Brenton, J., Caldas, C., Davies, J., Harris, S., Maccallum, P.: CancerGrid: Developing open standards for clinical cancer informatics. In: UK E-Science All Hands Meeting (2005)
3. Brenton, J., Davies, J., Gibbons, J., Harris, S.: Accelerating cancer research using semantics-driven technology. In: Microsoft eScience Workshop (December 2008)
4. Brooks Jr., F.P.: The Mythical Man-Month. Addison-Wesley (1975)
5. Calinescu, R.: Model-based SOA generation for cancer clinical trials. In: Skar, L.A., Bjerkestrand, A.A. (eds.) OOPSLA Workshop on Service-Oriented Architectures, Portland, Oregon, pp. 57–71 (2006)
6. Calinescu, R., Harris, S., Gibbons, J., Davies, J., Toujilov, I., Nagl, S.: Model-driven architecture for cancer research. In: Software Engineering and Formal Methods, pp. 59–68 (September 2007)
7. CancerGrid website, `http://www.cancergrid.org/`

8. CancerGrid metadata registry (cgMDR), http://www.cancergrid.org/index.php?option=com_content&id=8:mdrarticle
9. Cong, G., Fan, W., Jia, X., Ma, S.: PRATA: A system for XML publishing, integration and view maintenance. In: UK e-Science All Hands Meeting, pp. 432–435 (2006)
10. Crichton, C., Davies, J., Gibbons, J., Harris, S., Shukla, A.: Semantic frameworks for e-Government. In: Pardo, T., Janowski, T. (eds.) International Conference on Theory and Practice of Electronic Governance (ICEGOV), pp. 30–39 (2007)
11. Crichton, C., Davies, J., Gibbons, J., Harris, S., Shukla, A., Tsui, A.: Semantics-driven development for electronic government applications. In: HICSS Workshop on Electronic Government (2009)
12. Crichton, C., Davies, J., Gibbons, J., Harris, S., Tsui, A., Brenton, J.: Metadata-driven software for clinical trials. In: ICSE Workshop on Software Engineering in Health Care. IEEE (May 2009)
13. Davies, J., Gibbons, J., Harris, S., Metz, J., Pollard, A.J., Snape, M.: Model-driven support for a vaccine study in Kathmandu. In: Microsoft eScience Workshop (2009)
14. Davies, J., Gibbons, J., Harris, S., Warzel, D.: Evolving health informatics: Semantic frameworks and metadata-driven architectures. In: Microsoft eScience Workshop (2008)
15. Davies, J., Harris, S., Crichton, C., Shukla, A., Gibbons, J.: Metadata standards for semantic interoperability in electronic government. In: International Conference on Theory and Practice of Electronic Governance (2008)
16. Earl, H.: Neo-tAnGo: A neoadjuvant study of sequential epirubicin + cyclophosphamide and paclitaxel ± gemcitabine in the treatment of high risk early breast cancer with molecular profiling, proteomics and candidate gene analysis (2007) iSRCTN 78234870, http://public.ukcrn.org.uk/search/StudyDetail.aspx?StudyID=1229
17. Fegan, G.W., Lang, T.A.: Could an open-source clinical trial data-management system be what we have all been looking for? PLoS Medicine 5(3) (March 2008)
18. Fowler, M.: Domain Specific Languages. Addison Wesley (2010)
19. Harris, S., Calinescu, R.: CancerGrid clinical trials model 1.1. Tech. Rep. MRC/1.4.1.3, CancerGrid (2006), http://www.cancergrid.org/index.php?option=com_remository&func=fileinfo&id=185
20. ISO/IEC JTC1 SC32 WG2: ISO/IEC 11179, Information technology—metadata registries, http://metadata-standards.org/11179/
21. Kush, R.: Can the protocol be standardised? Tech. rep., Clinical Data Interchange Standards Consortium (2006)
22. McLaren, I., Wicks, T.: Developing generative frameworks using XML. In: Automated Software Engineering, pp. 368–372 (2001)
23. Mellor, S.J., Balcer, M.: Executable UML: A Foundation for Model-Driven Architecture. Addison-Wesley (2002)
24. Microsoft: InfoPath website, http://office.microsoft.com/en-us/infopath/
25. Microsoft: XML Schema Definition Tool (Xsd.exe) (2009), http://msdn.microsoft.com/en-us/library/x6c1kb0sVS.80.aspx
26. Miller, J., Mukerji, J.: Model driven architecture: A technical perspective. Tech. Rep. ormsc/2001-07-01, Object Management Group (July 2001)
27. Moher, D., Schulz, K., Altman, D.G.: The CONSORT statement: Revised recommendations for improving the quality of reports of parallel-group randomised trials. The Lancet, 357 (April 2001)

28. Poole, C., Earl, H.: NEAT: National breast cancer study of epirubicin plus CMF versus classical CMF adjuvant therapy (2001) iSRCTN 42625759, http://public.ukcrn.org.uk/search/StudyDetail.aspx?StudyID=643
29. Poole, C., Howard, H., Dunn, J.: tAnGo: A phase III randomized trial of gemcitabine in paclitaxel-containing, epirubicin-based adjuvant chemotherapy for women with early stage breast cancer (2004) iSRCTN 51146252, http://public.ukcrn.org.uk/search/StudyDetail.aspx?StudyID=661
30. Sierra, J.L., Fernández-Valmayor, A., Fernández-Manjón, B.: A document-oriented paradigm for the construction of content-intensive applications. Computer Journal 49(5), 562–584 (2006)
31. Steele, G.L.: Objects have not failed, position statement for OOPSLA panel (2002)
32. Stell, A., Sinnott, R., Ajayi, O.: Supporting the clinical trial recruitment process through the grid. In: UK e-Science All Hands Meeting, pp. 61–68 (2006)
33. Sullivan, L.: The tall office building artistically considered. Lippincott's Magazine (March 1896)
34. University of Birmingham School of Medicine: How Birmingham researchers are taking a measured look at medical treatments. Medlines 4 (July 1997), http://www.publications.bham.ac.uk/medlines/1997b/a_medpg3.htm
35. US National Cancer Institute: caCORE software development kit (2006), http://ncicb.nci.nih.gov/infrastructure/cacoresdk
36. US National Cancer Institute: Cancer Biomedical Informatics Grid (2006), https://cabig.nci.nih.gov/
37. Wong, P.Y.H., Gibbons, J.: A process-algebraic approach to workflow specification and refinement. In: Software Composition (2007)
38. Wong, P.Y.H., Gibbons, J.: On Specifying and Visualising Long-Running Empirical Studies. In: Vallecillo, A., Gray, J., Pierantonio, A. (eds.) ICMT 2008. LNCS, vol. 5063, pp. 76–90. Springer, Heidelberg (2008)
39. Wong, P.Y.H., Gibbons, J.: Formalisations and applications of BPMN. Science of Computer Programming 76, 633–650 (2011)
40. Wong, P.Y.H., Gibbons, J.: Property specifications for workflow modelling. Science of Computer Programming 76(10), 942–967 (2011)

Declarative Modelling and Safe Distribution of Healthcare Workflows

Thomas Hildebrandt[1], Raghava Rao Mukkamala[1], and Tijs Slaats[1,2,⋆]

[1] IT University of Copenhagen,
Rued Langgaardsvej 7, 2300 Copenhagen, Denmark
{hilde,rao,tslaats}@itu.dk
http://www.itu.dk

[2] Exformatics A/S, 2100 Copenhagen, Denmark

Abstract. We present a formal technique for safe distribution of workflow processes described declaratively as nested Dynamic Condition Response (DCR) Graphs and apply the technique to a distributed healthcare workflow. Concretely, we provide a method to synthesize from a nested DCR Graph and any distribution of its atomic events a set of local process graphs communicating by shared events, such that the distributed execution of the local processes is equivalent to executing the original process. The technique extends our recent work on safe distribution of non-nested DCR Graphs applied to cross-organizational case management. The main contributions of the present paper is to adapt the technique to allow for nested processes and milestones and to apply it to a healthcare workflow identified in a previous field study at Danish hospitals. We also provide a new formalization of the semantics of DCR Graphs which highlights its declarative nature.

1 Introduction

The overall goal of the interdisciplinary Trustworthy Pervasive Healthcare Services (TrustCare) project [16] is to develop a foundation for trustworthy IT-supported healthcare workflows. Healthcare workflows involve coordination of a heterogeneous set of professionals, patients, organizations and sectors, and must be able to adapt to inevitable changes of treatment processes and organization of the work [23,40]. This challenges traditional workflow management systems using an imperative process modeling language such as Business Process Model and Notation (BPMN) [35] in which the control flow is modeled explicitly. Typically a flow diagram will only cover the normal flow and a few possible exceptionally flows, leading to rigid and inflexible, over-specified workflows. Declarative process languages, allowing any flow that fulfills the specified constraints, have been suggested by a number of researchers as being more appropriate for representing workflow processes requiring a high degree of flexibility [8–10,33,43].

⋆ Authors listed alphabetically. This research is supported by the Danish Research Agency through the Trustworthy Pervasive Healthcare Services project (grant #2106-07-0019, www.trustcare.eu) and the Computer Supported Mobile Adaptive Business Processes project (grant #274-06-0415, www.cosmobiz.dk).

The TrustCare project combines research in pervasive user-interfaces [2,3] with research in formal logic and domain specific process languages, taking as starting point [31] the declarative workflow process language developed and used by Resultmaker, the industrial partner of the project.

The present paper focuses on and extends our work on formal process languages, in particular the development of a basic formal declarative workflow language called Dynamic Condition Response Graphs (DCR Graphs) introduced in [17] and extended to allow nested (i.e. hierarchical) process definitions in [19,20] and a new milestone relation between activities. In [21] we have shown how, given a (non-nested) DCR Graph describing a global, collaborative process and any distribution of the activities, to derive a set of local DCR Graphs corresponding to the activity distribution and achieving the same global behavior by synchronously communicating the relevant events between the local processes. The main new contributions of the present paper is to adapt the distribution technique given in [21] to allow for nested processes and the new milestone relation between activities as introduced in [20] and demonstrate the use of the technique on an oncology healthcare workflow previously identified during a field study at Danish hospitals [26]. The workflow was described in loc. cit. using an early formalization of the Resultmaker workflow process language. Another contribution of the present paper is to formalize the workflow process as a nested DCR Graph. Finally, we also provide a new presentation of the formal semantics of DCR Graphs that highlights the declarative nature. The present paper extends the results presented in the pre-proceedings of FHIES 2011 [22] which omitted the primitives of DCR Graphs for dynamically changing the set of included activities in the workflow.

The intention of the oncology healthcare workflow is to illustrate two important kinds of flexibility appearing in healthcare workflows: 1) The need to reconsider a previous activity if its validity is questioned at a later stage by a co-worker and 2) The need for distribution of collaborative tasks and ability to tailor this distribution to local conditions (e.g. the size and organization of work within a hospital). These needs have also been identified during field studies of case management processes [19] and appears to be generally relevant for knowledge work processes and not only healthcare workflows.

The rest of the paper is structured as follows. In Sec. 1.1 ending the introduction we briefly discuss related work. We present the oncology workflow example in Sec. 2 as a nested Dynamic Condition Response (DCR) Graph, describing the semantics informally. In Sec. 3 we recall the formal definition of nested DCR Graphs and provide a new presentation of their semantics. We then in Sec. 4 formalize the distribution technique and exemplify it on the oncology workflow. Finally we conclude in Sec. 5.

1.1 Related Work

The problem of verifying the correctness of cross-organizational workflows described as variants of Petri nets has been studied in [1, 25, 27, 39, 41, 42, 46] and models of global behavior based on conversations among participating services have been studied in [4,5,14,36,48,49]. A technique to partition a composite web service using program analysis was studied in [34] and using a similar approach, [24] explored decomposition of a business process modeled in BPEL, primarily focussing on P2P interactions . Using a formal approach based on I/O automata representing the services, the authors in [29] have studied the problem of synthesizing a decentralized choreography strategy, that will have optimal overhead of service composition in terms of costs associated with each interaction.

The derivation of descriptions of local components from a global model has also been researched in the work on structured communication-centred programming for web services by Carbone, Honda and Yoshida [6]. The work formalizes the core of WS-CDL as the global process calculus and defines a formal theory of end-point projections projecting the global process calculus to abstract descriptions of the behavior of each of the local "end-points" given as pi-calculus processes typed with session types.

A methodology for deriving process descriptions from a business contract formalized in a formal contract language was studied in [28], while [38] proposes an approach to extract a distributed process model from a collaborative business process. In [12,13], the authors have proposed a technique for the flexible decentralization of a process specification with necessary synchronization between the processing entities using dependency tables. In [7,15,32] foundational work has been made on synthesizing distributed transition systems from global specification for the models of synchronous product and asynchronous automata [50].

The formalisms discussed above are all confined to imperative modeling languages such as Petri nets, workflow/open nets and automata based languages. To the best of our knowledge, very few works exist on distributed cross-organizational workflows which consider declarative modeling languages and none where both the global and local processes are given declaratively using the same formalism. In [11], Fahland has studied synthesizing declarative workflows expressed in DecSerFlow [45] by translating to Petri nets. Only a predefined set of DecSerFlow constraints are used in the mapping to the Petri nets patterns, so this approach has a limitation with regards to the extensibility of the DecSerFlow language. On the other hand, in [30] Montali has studied the composition of ConDec [44] models with respect to conformance with a given choreography, based on the compatibility of the local ConDec models. But his study was limited to only composition of local models, whereas the problem of splitting a global model into local models has not been studied.

2 Distributed Declarative Healthcare Workflows by Example

In Fig. 1 below we show the graphical representation of the nested Dynamic Condition Response Graph formalizing a variant of the oncology workflow studied in [26]. In this section we informally describe the formalism and the distribution technique formalized in the rest of the paper using the example workflow. For details of the field study and the workflow we refer the reader to [26].

The boxes denote *activities* (also referred to as events in the following sections). Administer medicine is a *nested* activity having sub activities give medicine and trust. Give medicine is an *atomic* activity, i.e. it has no sub activities. Trust is again a nested activity having sub activities sign nurse 1 and sign nurse 2. Finally, medicine preparation is a nested activity having seven sub activities dealing with the preparation of medicine. An activity may be either included or excluded; the latter are drawn as dashed boxes as e.g. the edit and cancel activities.

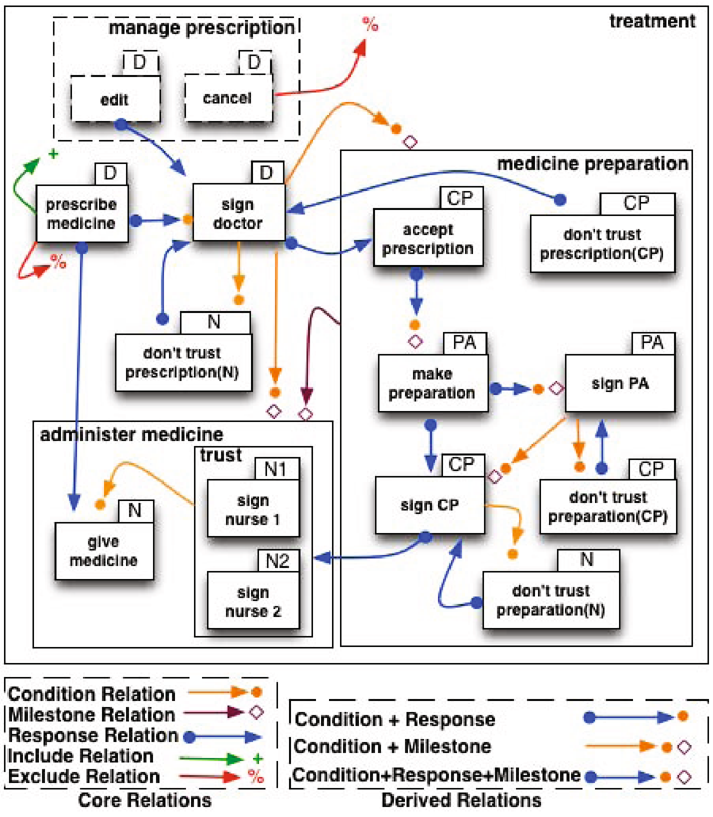


Fig. 1. Oncology Workflow as a nested DCR Graph

A *run* of the workflow consists of a (possibly infinite) sequence of executions of atomic activities. (A nested activity is considered executed when all its sub activities are executed). An activity can be executed any number of times during a run, as long as the activity is included and the constraints for executing it are satisfied, in which case we say the activity is *enabled*.

The constraints and dynamic exclusion and inclusion are expressed as five different core relations between activities represented as arrows in the figure above: The *condition relation*, the *response relation*, the *milestone relation*, the *include relation*, and the *exclude relation*.

The condition relation is represented by an orange arrow with a bullet at the arrow head. E.g. the condition relation from the activity sign doctor to the activity don't trust prescription(N) means that sign doctor must have been executed at least once before the activity don't trust prescription(N) can be executed.

The response relation is represented by a blue arrow with a bullet at its source. E.g. the response relation from the activity prescribe medicine to the activity give medicine means that the latter must be executed (at some point) after (any execution of) the activity prescribe medicine. We say that a workflow is in a *completed* state if all such response constraints have been fulfilled (or the required response activity is excluded). However, note that a workflow may be continued from a completed state and change to a non-completed state if an activity is executed that requires another response or includes an activity which has not been executed since it was last required as a response. Also note that the response constraint may cause some infinite runs to never pass through a complete state if the executed activities keep triggering new responses. In the following section we make precise when such infinite runs can be regarded as a complete execution.

The third core relation used in the example is the *milestone relation* represented as a dark red arrow with a diamond at the arrow head. The milestone relation was introduced in [20] jointly with the ability to nest activities. A relation to and/or from a nested activity simply unfolds to relations between all sub activities. A milestone relation from a nested activity to another activity then in particular means that the entire nested activity must be in a completed state before that activity can be executed. E.g. medicine preparation is a milestone for the activity administer medicine, which means that none of the sub activities of administer medicine can be carried out if any one of the sub activities of medicine preparation is included and has not been executed since it was required as a response.

Two activities can be related by any combination of these relations. In the graphical notation we have employed some shorthands, e.g. indicating the combination of a condition and a response relation by an arrow with a bullet at both ends.

Finally, DCR Graphs allow two relations for dynamic exclusion and dynamic inclusion of activities represented as a green arrow with a plus at the arrow head and a red arrow with a minus at the arrow head respectively. The exclusion relation is used in the example between the cancel activity and the treatment activity. Since all other activities in the workflow are sub activities of the treatment activity this means that all activities are excluded if the cancel activity is executed. The inclusion relation is used between the prescribe medicine activity and the manage prescription activity.

The run-time state of a nested DCR Graph can be formally represented as a pair $(\mathsf{Ex}, \mathsf{Re}, \mathsf{In})$ of sets of atomic activities (referred to as the *marking* of the graph). The set Ex is the set of atomic activities that have been executed at least once during the run. The set Re is the set of atomic activities that, if included, are required to be executed at least one more time in the future as the result of a response constraint (i.e. they are pending responses). Finally, the set In denotes the currently included activities.

The set Ex thus may be regarded as a set of completed activities, the set Re as the set of activities on the to-do list and the set In as the activities that are currently relevant for the workflow.

Note that an activity may be completed once and still be on the to-do list, which simply means that it must be executed (completed) again. This makes it very simple to model the situation where an activity needs to be (re)considered as a response to the execution of an activity. In the oncology example this is e.g. the case for the response relation between the don't trust prescription(N) activity (representing that a nurse reports that he doesn't trust the prescription) and the sign doctor activity. The effect is that the doctor is asked to reconsider her signature on the prescription. In doing that she may or may not decide to change the prescription, i.e. execute prescribe medicine again.

We indicate the marking graphically by adding a check mark to every atomic activity that has been executed (i.e. is included in the set Ex of the marking), an exclamation mark to every atomic activity which, if included, is required to be executed at least once more in the future (i.e. is included in the set Re), and making a box dashed if the activity is not included (i.e. is not included in the set In of the marking). In Fig. 1 we have shown an example marking where prescribe medicine has been executed. This has caused manage prescription and its sub activities edit and cancel to be included, and sign doctor and give medicine to be required as responses, i.e the two activities are included in the set Re of the marking (on the to-do list).

As described above, an activity can be executed if it is enabled. Sign doctor is enabled for execution in the example marking, since its only condition (prescribe medicine) has been executed and it has no milestones. Give medicine on the other hand is not enabled since it has the (nested) activity trust as condition, which means that all sub activities of trust (sign nurse 1 and sign nurse 2) must be executed before give medicine is enabled. Also, both give medicine and trust are sub activities of administer medicine which further has sign doctor as condition and milestone, and medicine preparation as milestone. The condition relation from sign doctor means that the prescription must be signed before the medicine can be administered. The milestone relations means that the medicine can not be given as long as sign doctor or any of the sub activities of medicine preparation is on the to-do list (i.e. in the set Re of pending responses).

Most activities should only be available to a subset of the users of the workflow system. For this reason the commercial implementation of the workflow management system provided by Resultmaker employs a role based access control, assigning to every atomic activity a finite set of roles and assigning to every role a set of access rights controlling if the activity is invisible or visible to users fulfilling the role. If an activity is visible it is specified wether the role is allowed to execute the activity or not. Users are either statically (e.g. by login) or dynamically assigned to roles (e.g. by email invitation).

In the formalization presented in this paper, the assigned roles are given as part of the name of the activity. In the graphical representation we have shown the roles within small ”ears” on the boxes. In the example workflow we have the following different roles: Doctor (D), Controlling Pharmacist (CP), Pharmacist Assistant (PA) and Nurse (N). Hereto comes roles N1 and N2 which must dynamically be assigned to two different authorized persons (nurses or doctors). This is at present the only way to implement the constraint stating that two different authorized persons must sign the product prepared by the pharmacists before the medicine is administered to the patient. Future work will address less ad hoc ways to handle these kind of constraints between activities referring to the identify of users.

The commercial implementation is based on a centralized workflow manager controlling the execution of the entire, global workflow. However, workflows often span different units or departments within the organization, e.g. the pharmacy and the patient areas, or even cross boundaries of different organizations (e.g. different hospitals). In some situations it may be very relevant to execute the local parts of the workflow on a local (e.g. mobile) device without permanent access to a network, e.g. during preparation of the medicine in the pharmacy. Also, different organizations may want to keep control of their own parts of the workflow and not delegate the management to a central service. This motivates the ability to split the workflow into separate components, each only referring to the activities relevant for the local unit and being manageable independently of the other components.

The technique for distributing DCR Graphs introduced in [21] and extended in the present paper is a first step towards supporting this kind of splitting of workflow definitions. Given any division of activities on local units (assigning every activity to at least one unit) it describes how to derive a set of graphs, one for each unit, describing the local part of the workflow. Such a local process, referred to as a *projection* is again a DCR Graph. It includes the activities assigned to the unit but also the relevant *external* activities executed within other units for which an event must be sent to the local process when they are executed. An example of a projection relative to the activities assigned the doctor role (D) is given in Fig. 2(a) in Sec. 4. The diagram shows that the projection also includes the two external activities (indicated as double line boxes) don't trust prescription (N) and don't trust prescription (CP). These two activities, representing respectively a nurse and a controlling pharmacist reporting that the prescription is not trusted, are the only external activities that may influence the workflow of the doctor by requiring sign doctor as a response. Similarly, Fig. 2(b),2(c), and 2(d) shows projections corresponding to the nurse, controlling pharmacist, and pharmacist assistant roles. However, if for instance the roles of the controlling pharmacist and the pharmacist assistant are always assigned to the same persons one may instead choose to keep all these activities together in a unit. This can be obtained by simply projecting on all activities assigned either the CP or the PA role.

3 Nested Dynamic Condition Response Graphs

Dynamic Condition Response Graphs (DCR Graphs) [17] is both a generalization of labelled event structures [47] and the Process Matrix workflow model developed by Resultmaker, our industrial partner in the TrustCare research project. The DCR Graphs were extended in [20] to Nested DCR Graphs, supporting sub graphs and a new milestone relation, motivated by a case study of cross-organizational case management [19]. Further in [21], we have considered safe distribution of DCR Graphs without milestone relation where as in [18], we have defined projection and distribution for a restricted nested model Condition Response Graphs, simplified by not allowing the dynamic inclusion and exclusion. In the present paper, we will consider full version of Nested DCR Graphs with both milestone and dynamic inclusion/exclusion relations and provide distribution of nested DCR Graphs. We employ the following notations in the paper.

Notation: For a set A we write $\mathcal{P}(A)$ for the power set of A. For a binary relation $\rightarrow \subseteq A \times A$ and a subset $\xi \subseteq A$ of A we write $\rightarrow \xi$ and $\xi \rightarrow$ for the set $\{a \in A \mid (\exists a' \in \xi \mid a \rightarrow a')\}$ and the set $\{a \in A \mid (\exists a' \in \xi \mid a' \rightarrow a)\}$ respectively. Also, we write $\rightarrow^{-1}$ for the inverse relation. Finally, for a natural number k we write $[k]$ for the set $\{1, 2, \ldots, k\}$.

We then formally define nested dynamic condition response graph as follows.

Definition 1. *A Dynamic Condition Response Graph (DCR Graph) G is a tuple* $(\mathsf{E}, \mathsf{M}, \rightarrow\!\bullet, \bullet\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$*, where*

(i) E *is the set of* events *(or activities),*
(ii) $\mathsf{M} = (\mathsf{Ex}, \mathsf{Re}, \mathsf{In}) \in \mathcal{M}(G)$ *is the* marking, *for* $\mathcal{M}(G) =_{def} \mathcal{P}(\mathsf{E}) \times \mathcal{P}(\mathsf{E}) \times \mathcal{P}(\mathsf{E})$,
(iii) $\rightarrow\!\!\bullet \subseteq \mathsf{E} \times \mathsf{E}$ *is the* condition *relation,*
(iv) $\bullet\!\!\rightarrow \subseteq \mathsf{E} \times \mathsf{E}$ *is the* response *relation,*
(v) $\rightarrow\!\!\diamond \subseteq \mathsf{E} \times \mathsf{E}$ *is the* milestone *relation,*
(vi) $\rightarrow\!+, \rightarrow\!\% \subseteq \mathsf{E}\times\mathsf{E}$ *is the dynamic* include *relation and* exclude *relation, satisfying that* $\forall e \in \mathsf{E}. e \rightarrow\!+ \cap e \rightarrow\!\% = \emptyset$,
(vii) L *is the set of* labels,
(viii) $l : \mathsf{E} \rightarrow \mathcal{P}(\mathsf{L})$ *is a labeling function mapping events to sets of labels.*

A Nested Dynamic Condition Response Graph (Nested DCR Graph) G is a tuple $(\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\!\!\bullet, \bullet\!\!\rightarrow, \rightarrow\!\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$*, where*

(i) $(\mathsf{E}, \mathsf{M}, \rightarrow\!\!\bullet, \bullet\!\!\rightarrow, \rightarrow\!\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$ *is a DCR Graph,*
(ii) $\rhd : \mathsf{E} \rightharpoonup \mathsf{E}$ *is a partial function mapping an event to its super-event (if defined),*
(iii) $\mathsf{M} \in \mathcal{P}(\mathsf{atoms}(\mathsf{E})) \times \mathcal{P}(\mathsf{atoms}(\mathsf{E})) \times \mathcal{P}(\mathsf{atoms}(\mathsf{E}))$*, where* $\mathsf{atoms}(\mathsf{E}) = \mathsf{E} \backslash \{e \in \mathsf{E} \mid \exists e' \in \mathsf{E}. \rhd (e') = e\}$ *is the set of atomic events.*

We write $e \rhd e'$ *if* $e' = \rhd^k(e)$ *for* $0 < k$ *and write* $e \trianglerighteq e'$ *if* $e \rhd e'$ *or* $e = e'$*, and* $e \trianglelefteq e'$ *if* $e' \rhd e$ *or* $e = e'$*. We require that the resulting relation,* $\trianglerighteq \subset E \times E$*, referred to as the nesting relation, is a well founded partial order. We also require that the nesting relation is consistent with respect to dynamic inclusion/exclusion in the following sense: If* $e \rhd e'$ *or* $e' \rhd e$ *then* $e \rightarrow\!+ \cap e' \rightarrow\!\% = \emptyset$.

We already introduced the graphical notation for Nested DCR Graphs by example in the previous section. We will not write out the complete formal specification. The events are all boxes, e.g. $\mathsf{E} = \{\mathsf{treatment}, \mathsf{manage\ prescription}, \mathsf{prescribe\ medicine}, ...\}$, the nesting relation captures the inclusion of boxes, e.g. $\rhd(e) = \mathsf{administer\ medicine}$, if $e \in \{\mathsf{give\ medicine}, \mathsf{trust}\}$ and $\rhd(e) = \mathsf{trust}$, if $e \in \{\mathsf{sign\ nurse1}, \mathsf{sign\ nurse2}\}$ and so forth. The inital marking is the triple $M = (\emptyset, \emptyset, \mathsf{E}\backslash\{\mathsf{manage\ prescription}, \mathsf{edit}, \mathsf{cancel}\})$, meaning no events have been executed, no events are initially required as responses and all events except the events $\{\mathsf{manage\ prescription}, \mathsf{edit}, \mathsf{cancel}\}$ are included. The condition relation includes e.g. the pairs $\mathsf{sign\ doctor} \rightarrow\!\!\bullet \mathsf{don't\ trust\ prescription(N)}$ and $\mathsf{trust} \rightarrow\!\!\bullet \mathsf{give\ medicine}$, the response relation includes e.g. the pairs $\mathsf{prescribe\ medicine} \bullet\!\!\rightarrow \mathsf{sign\ doctor}$ and $\mathsf{edit} \bullet\!\!\rightarrow \mathsf{sign\ doctor}$, the milestone relation includes e.g. the pairs $\mathsf{sign\ doctor} \rightarrow\!\!\diamond \mathsf{administer\ medicine}$ and $\mathsf{sign\ PA} \rightarrow\!\!\diamond \mathsf{sign\ CP}$, the dynamic inclusion relation includes the single pair $\mathsf{prescribe\ medicine} \rightarrow\!+ \mathsf{manage\ prescription}$ and the dynamic exclusion includes exactly the two pairs $\mathsf{prescribe\ medicine} \rightarrow\!\% \mathsf{prescribe\ medicine}$ and $\mathsf{cancel} \rightarrow\!\% \mathsf{treatment}$. We take as labels pairs of action names and roles, i.e. the set of labels L includes e.g. the pairs $(\mathsf{edit}, \mathsf{D})$, $(\mathsf{cancel}, \mathsf{D})$, $(\mathsf{give\ medicine}, \mathsf{N})$, and $(\mathsf{sign\ PA}, \mathsf{PA})$. Super events with no role assigned such as $\mathsf{manage\ prescription}$ are assigned the empty set of labels.

Note that the labels of events consist of the name of the event and a role which defines who can execute that event. In our implementation every event can be assigned any number of roles and every user of the system can have multiple roles. A user can then execute an event if she has at least one role that is assigned to the event.

To define the execution semantics for Nested DCR Graphs we first define how to flatten a nested graph to the simpler DCR Graph. Essentially, all relations to and/or from nested events are extended to sub events, and then only the atomic events are preserved.

Definition 2. *For a Nested DCR Graph* $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\!\bullet, \bullet\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$ *define the underlying flat Dynamic Condition Response Graph as*

$$G^{\flat}_{l_f} = (\mathsf{atoms}(\mathsf{E}), \mathsf{M}, \rightarrow\!\bullet^{\flat}, \bullet\!\rightarrow^{\flat}, \rightarrow\!\diamond^{\flat}, \rightarrow\!+^{\flat}, \rightarrow\!\%^{\flat}, \mathsf{L}, l)$$

where $rel^{\flat} = \trianglerighteq rel \trianglelefteq$ *for some relation* $rel \in \{\rightarrow\!\bullet, \bullet\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%\}$.

It is easy to see from the definition that the underlying DCR Graph has at most as many events as the nested graph and that the size of the relations may increase by an order of n^2 where n is the number of atomic events.

Below we give a new presentation of the semantics of DCR Graphs stressing its declarative nature.

First we formalize in Def. 3 that an event e of a (flat) DCR Graph is enabled when it is included in current marking ($e \in \mathsf{In}$), all the included events that are conditions for it are in the set of executed events (i.e. $(\mathsf{In} \cap \rightarrow\!\bullet\, e) \subseteq \mathsf{Ex}$) and none of the included events that are milestones for it are in the set of pending response events (i.e. $(\mathsf{In} \cap \rightarrow\!\diamond e) \subseteq \mathsf{E} \backslash \mathsf{Re}$).

Definition 3. *For a Dynamic Condition Response Graph* $G = (\mathsf{E}, \mathsf{M}, \rightarrow\!\bullet, \bullet\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$*, and* $\mathsf{M} = (\mathsf{Ex}, \mathsf{Re}, \mathsf{In})$ *we define that an event* $e \in \mathsf{E}$ *is* enabled*, written* $G \vdash e$*, if* $e \in \mathsf{In}$*,* $(\mathsf{In} \cap \rightarrow\!\bullet\, e) \subseteq \mathsf{Ex}$ *and* $(\mathsf{In} \cap \rightarrow\!\diamond e) \subseteq \mathsf{E} \backslash \mathsf{Re}$.

Def. 4 below then defines the change of the marking when an enabled event is executed: First the event is added to the set of executed events and removed from the set of pending responses. Then all events that are a response to the event are added to the set of pending responses. Note that if an event is a response to itself, it will remain in the set of pending responses after execution. Similarly, the included events set will updated by adding all the events that are included by the event and by removing all the events that are excluded by the event.

Definition 4. *For a Dynamic Condition Response Graph* $G = (\mathsf{E}, \mathsf{M}, \rightarrow\!\bullet, \bullet\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$*, where* $\mathsf{M} = (\mathsf{Ex}, \mathsf{Re}, \mathsf{In})$ *and an enabled event* $G \vdash e$ *, the result of executing* e *is a Dynamic Condition Response Graph* $G = (\mathsf{E}, \mathsf{M}', \rightarrow\!\bullet, \bullet\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$*, where* $\mathsf{M}' = (\mathsf{Ex}, \mathsf{Re}, \mathsf{In}) \oplus_G e =_{def} \big(\mathsf{Ex} \cup \{e\}, (\mathsf{Re} \setminus \{e\}) \cup e \bullet\!\rightarrow, (\mathsf{In} \cup e\!\rightarrow\!+) \setminus e\!\rightarrow\!\%\big)$.

We now define the semantics for Nested DCR Graph by using the corresponding flat graph.

Definition 5. *For a Nested Condition Response Graph* $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\!\bullet, \bullet\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$*, where* $\mathsf{M} = (\mathsf{Ex}, \mathsf{Re}, \mathsf{In})$ *we define that* $e \in \mathsf{atoms}(\mathsf{E})$ *is enabled, written* $G \vdash e$*, if* $G^{\flat} \vdash e$*. Similarly, the result of executing* $G \vdash e$ *is defined as:* $(\mathsf{Ex}, \mathsf{Re}, \mathsf{In}) \oplus_{G^{\flat}} e$.

As an example, in the intial marking $M = (\emptyset, \emptyset, \mathsf{E}\backslash\{\mathsf{manage\ prescription}, \mathsf{edit}, \mathsf{cancel}\})$ we have that $G \vdash \mathsf{prescribe\ medicine}$, i.e. the event prescribe medicine is enabled.

After executing prescribe medicine the new marking $M' = M \oplus_G \mathsf{prescribe\ medicine}$ $= (\{\mathsf{prescribe\ medicine}\}, \{\mathsf{sign\ doctor}, \mathsf{give\ medicine}\}, \mathsf{E}\backslash\{\mathsf{prescribe\ medicine}\})$. That is, prescribe medicine is added to the set of executed events, and sign doctor and give medicine are added to the set of pending responses, because prescribe medicine $\bullet\!\!\rightarrow$ sign doctor and prescribe medicine $\bullet\!\!\rightarrow$ give medicine}. The event prescribe medicine is removed from the set of included events because prescribe medicine $\rightarrow\!\%$ prescribe medicine. The events $\{\mathsf{manage\ prescription}, \mathsf{edit}, \mathsf{cancel}\}$ are included since prescribe medicine $\rightarrow\!+$ manage prescription, and the inclusion relation is "flattened" to include also prescribe medicine $\rightarrow\!+$ edit and prescribe medicine $\rightarrow\!+$ cancel.

From the definition of enabling and execution above we can construct a labelled transition semantics for a nested DCR Graphs, with acceptance conditions for finite and infinite computations.

Definition 6. *For a nested DCR Graph* $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\!\bullet, \bullet\!\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$ *we write* $G \xrightarrow{(e,a)} G'$ *if* $G' = (\mathsf{E}, \rhd, \mathsf{M}', \rightarrow\!\bullet, \bullet\!\!\rightarrow, \rightarrow\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$*,* $G \vdash e$*,* $a \in l(e)$*, and* $\mathsf{M}' = \mathsf{M} \oplus_{G^\flat} e$*. We then define the corresponding labelled transition system* $TS(G)$ *to be the tuple* $(\mathcal{G}_{\mathsf{E}\times\mathsf{L}}, G, \mathsf{E} \times \mathsf{L}, \rightarrow \subseteq \mathcal{G}_{\mathsf{E}\times\mathsf{L}} \times \mathsf{E} \times \mathsf{L} \times \mathcal{G}_{\mathsf{E}\times\mathsf{L}})$ *where* $\mathcal{G}_{\mathsf{E}\times\mathsf{L}}$ *is the set of all nested DCR Graphs with events in* E *and labels in* L*.*

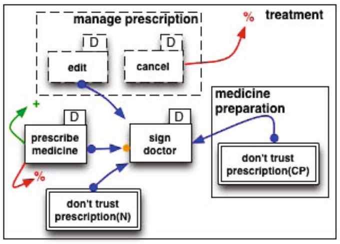
We define a run $a_0, a_1, \ldots$ *of a nested DCR Graph G to be a sequence of labels of a sequence of transitions* $G_i \xrightarrow{(e_i,a_i)} G_{i+1}$*. starting from* $G_0 = G$*. Assuming the marking of* G_i *is* $(\mathsf{Ex}_i, \mathsf{Re}_i, \mathsf{In}_i)$*, define a run to be accepting if for the underlying sequence of transitions it holds that* $\forall i \geq 0, e \in \mathsf{Re}_i.\exists j \geq i.(e = e_j \vee e \notin \mathsf{In}_j)$*. In words, a run is accepting if every required response event happens at some later stage or become excluded.*

4 Projections and Distributed Execution

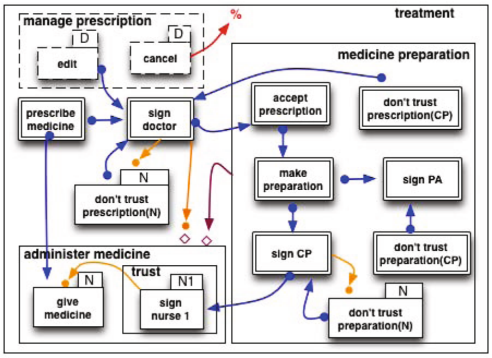
In Sec. 4.1 below we define the notion of projection of a nested DCR Graphs, restricting the graph to a subset of the events, and in Sec. 4.2 we define the technique for distributing a nested DCR Graph as a set of local nested DCR Graphs obtained as projections and communicating by notifications of event executions.

4.1 Projections

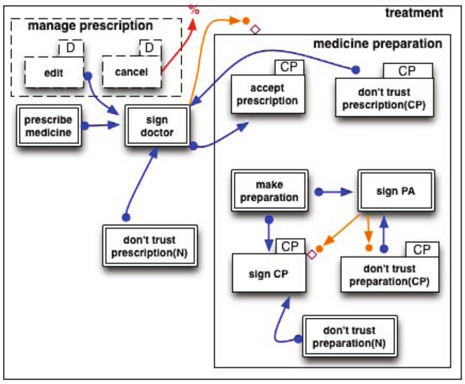
A nested DCR Graph G is projected with respect to a *projection parameter* $\delta = (\delta_{\mathsf{E}}, \delta_{\mathsf{L}})$, where $\delta_{\mathsf{E}} \subseteq \mathsf{E}$ is a subset of the events of G satisfying that $\rhd(\delta_{\mathsf{E}}) \subseteq \delta_{\mathsf{E}}$, i.e. the subset is closed under the super event relation, and $\delta_{\mathsf{L}} \subseteq \mathsf{L}$ is a subset of the labels. The intuition is that the graph is restricted to only those events and relations that are relevant for the execution of events in δ_{E} and the labeling is restricted to the set δ_{L}. The technical difficulty is to infer the events and relations not in δ_{E}, referred to as *external events* below, that should be included in the projection because they influence the execution of the workflow restricted to the events in δ_{E}.



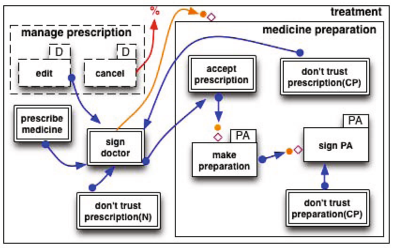
(a) Projection over *D*



(b) Projection over *N* and *N1*



(c) Projection over *CP*



(d) Projection over *PA*

Fig. 2. Oncology workflow projected with respect to different roles

Fig. 2 shows examples of projections of the oncology workflow with respect to different roles. For instance, Fig. 2(a) shows the projection with respect to the projection parameter $(\delta_{\mathsf{E}}, \delta_{\mathsf{L}})$ where δ_{E}={manage prescription, edit, cancel, prescribe medicine, sign doctor} and δ_{L}={(edit, D), (cancel, D), (prescribe medicine, D), (sign doctor, D)}. The two events don't trust prescription (N) and don't trust prescription (CP) shown with double line borders are external events included in the projected graph even though they don't appear in the projection parameter. It is interesting to note that the doctor only needs to be aware of these two activities carried out by other participants. In comparison, the projection over the roles for nurses (*N* and *N1*) contains all the events since they may influence (because of the milestone relations) the execution of the events with roles *N* and *N1*. In other words, the doctors can carry out workflows highly independent of the other activities while the nurses are dependent on any event carried out by the other roles.

The formal definition of projection for nested DCR Graphs is given in 7 below. It generalizes the definition of projection introduced in [21] for DCR Graphs to support nesting and milestones.

Definition 7. *If* $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\!\!\bullet, \bullet\!\!\rightarrow, \rightarrow\!\!\diamond, \rightarrow\!+, \rightarrow\!\%, \mathsf{L}, l)$ *then* $G_{|\delta} = (\mathsf{E}_{|\delta}, \rhd_{|\delta}, \mathsf{M}_{|\delta}, \rightarrow\!\!\bullet_{|\delta}, \bullet\!\!\rightarrow_{|\delta}, \rightarrow\!\!\diamond_{|\delta}, \rightarrow\!+_{|\delta}, \rightarrow\!\%_{|\delta}, \delta_{\mathsf{L}}, l_{|\delta})$ *is the projection of* G *with respect to* $\delta \subseteq \mathsf{E}$ *where:*

(i) $\mathsf{E}_{|\delta} = \rightarrow \delta_{\mathsf{E}}$*, for* $\rightarrow = \bigcup_{c \in C} c$*, and* $C = \{\mathsf{id}, \rightarrow\!\!\bullet^\flat, \bullet\!\!\rightarrow^\flat, \rightarrow\!\!\diamond^\flat, \rightarrow\!+^\flat, \rightarrow\!\%^\flat, \bullet\!\!\rightarrow^\flat\rightarrow\!\!\diamond^\flat, \rightarrow\!+^\flat\rightarrow\!\!\bullet^\flat, \rightarrow\!\%^\flat\rightarrow\!\!\bullet^\flat, \rightarrow\!+^\flat\rightarrow\!\!\diamond^\flat, \rightarrow\!\%^\flat\rightarrow\!\!\diamond^\flat\}$

(ii) $\rhd_{|\delta}(e) = \rhd(e)$*, if* $e \in \mathsf{E}_{|\delta}$

(iii) $l_{|\delta}(e) = \begin{cases} l(e) \cap \delta_{\mathsf{L}} & \textit{if } e \in \delta_{\mathsf{E}} \\ \emptyset & \textit{if } e \in \mathsf{E}_{|\delta} \backslash \delta_{\mathsf{E}} \end{cases}$

(iv) $\mathsf{M}_{|\delta} = (\mathsf{Ex}_{|\delta}, \mathsf{Re}_{|\delta}, \mathsf{In}_{|\delta})$ *where:*

  (a) $\mathsf{Ex}_{|\delta} = \mathsf{Ex} \cap \mathsf{E}_{|\delta}$

  (b) $\mathsf{Re}_{|\delta} = \mathsf{Re} \cap (\delta_{\mathsf{E}} \cup \rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}})$

  (c) $\mathsf{In}_{|\delta} = \mathsf{In} \cap \mathsf{E}_{|\delta}$

(v) $\rightarrow\!\!\bullet_{|\delta} = \rightarrow\!\!\bullet \cap((\rightarrow\!\!\bullet^\flat \delta_{\mathsf{E}}) \times \delta_{\mathsf{E}})$

(vi) $\bullet\!\!\rightarrow_{|\delta} = \bullet\!\!\rightarrow \cap((\bullet\!\!\rightarrow^\flat\rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}}) \times (\rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}})) \cup ((\bullet\!\!\rightarrow^\flat \delta_{\mathsf{E}}) \times \delta_{\mathsf{E}}))$

(vii) $\rightarrow\!\!\diamond_{|\delta} = \rightarrow\!\!\diamond \cap((\rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}}) \times \delta_{\mathsf{E}})$

(viii) $\rightarrow\!+_{|\delta} = \rightarrow\!+ \cap\Big(((\rightarrow\!+^\flat \delta_{\mathsf{E}}) \times \delta_{\mathsf{E}}) \cup ((\rightarrow\!+^\flat\rightarrow\!\!\bullet^\flat \delta_{\mathsf{E}}) \times (\rightarrow\!\!\bullet^\flat \delta_{\mathsf{E}}) \cup ((\rightarrow\!+^\flat\rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}}) \times (\rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}})\Big)$

(ix) $\rightarrow\!\%_{|\delta} = \rightarrow\!\% \cap\Big(((\rightarrow\!\%^\flat \delta_{\mathsf{E}}) \times \delta_{\mathsf{E}}) \cup ((\rightarrow\!\%^\flat\rightarrow\!\!\bullet^\flat \delta_{\mathsf{E}}) \times (\rightarrow\!\!\bullet^\flat \delta_{\mathsf{E}}) \cup ((\rightarrow\!\%^\flat\rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}}) \times (\rightarrow\!\!\diamond^\flat \delta_{\mathsf{E}})\Big)$

(*i*) defines the set of events in the projection as all events that has a relation pointing to an event in the set δ_{E}, where the relation is either the identity relation (i.e. it is an event in δ_{E}), one of the core relations (flattened) or the relations such as $\bullet\!\!\rightarrow^\flat\rightarrow\!\!\diamond^\flat$ which includes all events that triggers as a response some event that is a milestone to an event in δ_{E} or the relations that include/exclude conditions and milestones to an event in the set δ_{E}.

Events in $\mathsf{E}_{|\delta} \backslash \delta_{\mathsf{E}}$ are referred to as external events and will be included in the projection without labels, as can be seen from the definition of the labeling function in (*iii*). As we will formalize below, events without labels can not be executed by a user locally. However, when composed in a network containing other processes that can execute these events, their execution will be communicated to the process.

(*iv*) defines the projection of the marking: The executed and included event sets are simply restricted to the events in $\mathsf{E}_{|\delta}$. The responses are restricted to events in δ_{E} and events that have a milestone relation to an event in δ_{E} because these are the only responses that will affect the local execution of the projected graph. Note that these events will by definition be events in $\mathsf{E}_{|\delta}$ but may be external events.

Finally, (*v*) - (*ix*) state which relations should be included in the projection. For the events in δ_{E} all incoming relations should be included. Additionally response relations to events that are a milestone for an event in δ_{E} are included as well.

To define networks of communicating nested DCR Graphs and their semantics we use the following extension of a nested DCR Graph adding a new label to every event.

Definition 8. *For an DCR Graph* $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\bullet, \bullet\rightarrow, \rightarrow\diamond, \rightarrow+, \rightarrow\%, \mathsf{L}, l)$ *define* $G^{\varepsilon} = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\bullet, \bullet\rightarrow, \rightarrow\diamond, \rightarrow+, \rightarrow\%, \mathsf{L} \cup \{\varepsilon\}, l^{\varepsilon})$*, where* $l^{\varepsilon} = l(e) \cup \{\varepsilon\}$ *(assuming that* $\varepsilon \notin \mathsf{L}$*).*

We are now ready to state the key correspondence between global execution of events and the local execution of events in a projection.

Proposition 1. *Let* $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\bullet, \bullet\rightarrow, \rightarrow\diamond, \rightarrow+, \rightarrow\%, \mathsf{L}, l)$ *be a nested DCR Graph and* $G_{|\delta}$ *its projection with respect to a projection parameter* $\delta = (\delta_{\mathsf{E}}, \delta_{\mathsf{L}})$*. Then*

1. *for* $e \in \delta_{\mathsf{E}}$ *and* $a \in \delta_{\mathsf{L}}$ *it holds that* $G \xrightarrow{(e,a)} G'$ *if and only if* $G_{|\delta} \xrightarrow{(e,a)} G'_{|\delta}$,
2. *for* $e \notin \mathsf{E}_{|\delta}$ *it holds that* $G \xrightarrow{(e,a)} G'$ *implies* $G_{|\delta} = G'_{|\delta}$,
3. *for* $e \in \mathsf{E}_{|\delta}$ *it holds that* $G \xrightarrow{(e,a)} G'$ *implies* $(G_{|\delta})^{\varepsilon} \xrightarrow{(e,\varepsilon)} (G'_{|\delta})^{\varepsilon}$,

4.2 Distributed Execution

We are now ready to define networks of nested DCR Graphs and give the main technical theorem of the paper stating that a network of nested DCR Graphs obtained by projecting a nested DCR Graph G with respect to a covering vector of projection parameters has the same behavior as the original graph G.

Intuitively, a vector of projection parameters is covering if every event is included in at least one projection parameter and every label that is assigned to an event occurs at least once together with that event.

Definition 9. *We call a vector* $\Delta = (\delta_1, \ldots, \delta_k)$ *of projection parameters* covering *for some DCR Graph* $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\bullet, \bullet\rightarrow, \rightarrow\diamond, \rightarrow+, \rightarrow\%, \mathsf{L}, l)$ *if:*

1. $\bigcup_{i \in [k]} \delta_{\mathsf{E}i} = \mathsf{E}$ *and*
2. $(\forall a \in \mathsf{L}.\forall e \in \mathsf{E}.a \in l(e) \Rightarrow (\exists i \in [k].e \in \delta_{\mathsf{E}i} \wedge a \in \delta_{\mathsf{L}i})$

Definition 10. *We define a* network of DCR Graphs N *by the grammar*

$$N := G \mid N \| N$$

and let $\mathcal{N}_{\mathsf{E} \times \mathsf{L}}$ *be the set of all networks with events in* E *and labels in* L.

We write $\Pi_{i \in [n]} G_i$ *for* $G_1 \| G_2 \| \ldots \| G_n$*. We define the set of events of a network of graphs inductively by* $\mathcal{E}(G) = \mathsf{E}$ *and* $\mathcal{E}(N_1 \| N_2) = \mathcal{E}(N_1) \cup \mathcal{E}(N_2)$*. Similarly, we define the set of labels of a network of graphs inductively by* $\mathcal{L}(G) = \mathsf{L}$ *and* $\mathcal{L}(N_1 \| N_2) = \mathcal{L}(N_1) \cup \mathcal{L}(N_2)$.

Definition 11. *The transition semantics of networks of DCR Graphs are given by the following inference rules:*

input $$\frac{G_1^{\varepsilon} \xrightarrow{(e,\varepsilon)} G_2^{\varepsilon}}{G_1 \overset{\rhd e}{\rightsquigarrow} G_2}$$

sync input $$\frac{N_1 \overset{\rhd e}{\rightsquigarrow} N_1' \quad N_2 \overset{\rhd e}{\rightsquigarrow} N_2'}{N_1 \| N_2 \overset{\rhd e}{\rightsquigarrow} N_1' \| N_2'}$$

local input $$\frac{N_i \overset{\rhd e}{\rightsquigarrow} N_i' \quad e \notin \mathcal{E}(N_{1-i})}{N_0 \| N_1 \overset{\rhd e}{\rightsquigarrow} N_0' \| N_1} \quad N_{1-i} = N_{1-i}', i \in \{0,1\}$$

sync step $$\frac{N_i \xrightarrow{(e,a)} N_i' \quad N_{1-i} \overset{\rhd e}{\rightsquigarrow} N_{1-i}'}{N_0 \| N_1 \xrightarrow{(e,a)} N_0' \| N_1'} \quad i \in \{0,1\}$$

local step $$\frac{N_i \xrightarrow{(e,a)} N_i' \quad e \notin \mathcal{E}(N_{i-1})}{N_0 \| N_1 \xrightarrow{(e,a)} N_0' \| N_1} \quad N_{1-i} = N_{1-i}', i \in \{0,1\}$$

For a network of nested DCR Graphs N we define the corresponding transition system $TS(N)$ *by* $(\mathcal{N}_{\mathcal{EL}(N)}, N, \mathcal{EL}(N), \rightarrow \subseteq \mathcal{N}_{\mathcal{EL}(N)} \times \mathcal{EL}(N) \times \mathcal{N}_{\mathcal{EL}(N)})$ *where* $\mathcal{EL}(N) = \mathcal{E}(N) \times \mathcal{L}(N)$ *and the transition relation* $\rightarrow \subseteq \mathcal{N}_{\mathcal{EL}(N)} \times \mathcal{EL}(N) \times \mathcal{N}_{\mathcal{EL}(N)}$ *is defined by the inference rules above.*

We define a run $a_0, a_1, \ldots$ *of the transition system to be a sequence of labels of a sequence of transitions* $N_i \xrightarrow{(e_i,a_i)} N_{i+1}$ *starting from the initial network. We define a run for a network* $N = \Pi_{i \in [n]} G_i$ *to be accepting if for the underlying sequence of transitions it holds that* $\forall j \in [n], \forall i \geq 0, e \in \mathsf{Re}_{j,i}. \exists k \geq i.(e = e_k \vee e \notin \mathsf{In}_k)$*, where* $\mathsf{Re}_{j,i}$ *is the set of required responses in the* j*th nested DCR Graph in the network in the* i*th step of the run. In words, a run is accepting if every response event in a local nested DCR Graph in the network happens at some later state or becomes excluded.*

Thm. 1 below now states the correspondence between a nested DCR Graph and the network of nested DCR Graphs obtained from a covering projection.

Theorem 1. *For a nested DCR Graph G and a covering vector of projection parameters* $\Delta = (\delta_1, \ldots, \delta_n)$ *it holds that* $TS(G)$ *is bisimilar to* $TS(G_\Delta)$*, where* $G_\Delta = \Pi_{i \in [n]} G_{|\delta_i}$*. Moreover, a run is accepting in* $TS(G)$ *if and only if the bisimilar run is accepting in* $TS(G_\Delta)$*.*

The generality of the distribution technique given above allows for fine tuned projections where we select only a few events for a specific role and actor, but in most cases the parameter is likely to be chosen so that the projected graph shows the full responsibilities of a specific role or actor. A set of nested DCR Graphs can be maintained and executed in a distributed fashion, meaning that there is a separate implementation for every graph and that the execution of shared events is communicated between them. Through the distributed execution of projected graphs, nested DCR Graphs can be used as a (declarative) choreography model to the line of work (on typed imperative process models) in [6]: The original graph can be seen as the choreography, describing how the system as a whole should function, from which we project multiple end-points for individual roles or actors that can be implemented independently.

5 Conclusion and Future Work

We have presented a formal technique for safe distribution of collaborative, cross-organizational workflows modeled declaratively in the model of Nested Dynamic Condition Response (Nested DCR) Graphs [17,20]. The key difference between the present work and the related work surveyed in Sec. 1.1 is that Nested DCR Graphs is a declarative model while most previous work has focussed on imperative models. We have argued for the use of a declarative approach for flexible workflows of knowledge workers and exemplified the techniques on a small workflow identified during a previous field study at Danish hospitals [26]. The example is not supposed to demonstrate completeness of the technique but to capture two common examples of flexibility, namely the need to reconsider previous activities and the flexibility to distribute the execution of a workflow across different units within or across organizations.

The distribution technique presented is an extension of a method developed recently for flat DCR Graphs [21] to allow for nesting of events and the co-called milestone relations. This again allows us to apply the technique to the oncology workflow which we believe is an important new contribution in order to communicate the ideas better to people working within the healthcare domain.

A number of interesting questions are left for future work. We have implemented a prototype tool for design, simulation and verification (model checking via the SPIN and ZING model checkers) of DCR Graphs as reported on in [19]. These tools should be extended to nested graphs and the distribution technique should be implemented. This then leads to considering what can be achieved by performing verification of local components individually. We also aim to investigate how to support dynamic changes to local components, using the underlying idea of the distribution technique to determine what should be changed in other components when a local component is changed. Finally we are working on extending the model to allow for data and time to be represented and developing a prototype implementation integrated with the work on pervasive user interfaces carried out in the other track of the TrustCare project. This would allow us to carry out a larger demonstration project jointly with a hospital evaluating both the workflow modeling and the pervasive user interfaces. Along the same lines, it would be interesting to relate our work to the approach in the OpenKnowledge and Safe & Sound projects based on the Lightweight Coordination Calculus (LCC) [37].

Acknowledgements. We are grateful to the anonymous reviewers for valuable feedback.

References

1. van der Aalst, W.M.P., Weske, M.: The P2P Approach to Interorganizational Workflows. In: Dittrich, K.R., Geppert, A., Norrie, M. (eds.) CAiSE 2001. LNCS, vol. 2068, pp. 140–156. Springer, Heidelberg (2001)

2. Bardram, J.E., Bunde-Pedersen, J., Doryab, A., Sørensen, S.: CLINICAL SURFACES – Activity-Based Computing for Distributed Multi-Display Environments in Hospitals. In: Gross, T., Gulliksen, J., Kotzé, P., Oestreicher, L., Palanque, P., Prates, R.O., Winckler, M. (eds.) INTERACT 2009. LNCS, vol. 5727, pp. 704–717. Springer, Heidelberg (2009)
3. Bardram, J.E., Bunde-Pedersen, J., Soegaard, M.: Support for activity-based computing in a personal computing operating system. In: CHI 2006: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 211–220. ACM Press, New York (2006)
4. Bravetti, M., Zavattaro, G.: Contract Based Multi-party Service Composition. In: Arbab, F., Sirjani, M. (eds.) FSEN 2007. LNCS, vol. 4767, pp. 207–222. Springer, Heidelberg (2007)
5. Bravetti, M., Zavattaro, G.: A theory of contracts for strong service compliance. Mathematical. Structures in Comp. Sci. 19, 601–638 (2009)
6. Carbone, M., Honda, K., Yoshida, N.: Structured Communication-Centred Programming for Web Services. In: De Nicola, R. (ed.) ESOP 2007. LNCS, vol. 4421, pp. 2–17. Springer, Heidelberg (2007)
7. Castellani, I., Mukund, M., Thiagarajan, P.: Synthesizing Distributed Transition Systems from Global Specifications. In: Pandu Rangan, C., Raman, V., Sarukkai, S. (eds.) FST TCS 1999. LNCS, vol. 1738, pp. 219–231. Springer, Heidelberg (1999)
8. Chesani, F., De Matteis, P., Mello, P., Montali, M., Storari, S.: A Framework for Defining and Verifying Clinical Guidelines: A Case Study on Cancer Screening. In: Esposito, F., Raś, Z., Malerba, D., Semeraro, G. (eds.) ISMIS 2006. LNCS (LNAI), vol. 4203, pp. 338–343. Springer, Heidelberg (2006)
9. Chesani, F., Lamma, E., Mello, P., Montali, M., Storari, S., Baldazzi, P., Manfredi, M.: Computer-Based Medical Guidelines and Protocols: a Primer and Current Trends. In: Compliance Checking of Cancer-Screening Careflows: an Approach Based on Computational Logic. IOS Press (2008)
10. Chesani, F., Mello, P., Montali, M., Storari, S.: Testing Careflow Process Execution Conformance by Translating a Graphical Language to Computational Logic. In: Bellazzi, R., Abu-Hanna, A., Hunter, J. (eds.) AIME 2007. LNCS (LNAI), vol. 4594, pp. 479–488. Springer, Heidelberg (2007)
11. Fahland, D.: Towards analyzing declarative workflows. In: Autonomous and Adaptive Web Services (2007)
12. Fdhila, W., Godart, C.: Toward synchronization between decentralized orchestrations of composite web services. In: CollaborateCom 2009, pp. 1–10 (2009)
13. Fdhila, W., Yildiz, U., Godart, C.: A flexible approach for automatic process decentralization using dependency tables. In: International Conference on Web Services (2009)
14. Fu, X., Bultan, T., Su, J.: Realizability of conversation protocols with message contents. In: Proceedings of the IEEE International Conference on Web Services, ICWS 2004, pp. 96–103. IEEE Computer Society, Washington, DC (2004)
15. Heljanko, K., Stefanescu, A.: Complexity results for checking distributed implementability. In: Proceedings of the Fifth International Conference on Application of Concurrency to System Design, pp. 78–87 (2005)
16. Hildebrandt, T.: Trustworthy pervasive healthcare processes (TrustCare) research project (2008), http://www.trustcare.dk/
17. Hildebrandt, T., Mukkamala, R.R.: Declarative event-based workflow as distributed dynamic condition response graphs. In: Post Proceedings of International Workshop on Programming Language Approaches to Concurrency and Communication-cEntric Software, PLACES 2010 (2011)
18. Hildebrandt, T., Mukkamala, R.R., Slaats, T.: Declarative modelling and safe distribution of healthcare workflows. In: International Symposium on Foundations of Health Information Engineering and Systems, Johannesburg, South Africa (August 2011)

19. Hildebrandt, T., Mukkamala, R.R., Slaats, T.: Designing a cross-organizational case management system using dynamic condition response graphs. In: Proceedings of IEEE International EDOC Conference (2011) (to appear)
20. Hildebrandt, T., Mukkamala, R.R., Slaats, T.: Nested dynamic condition response graphs. In: Proceedings of Fundamentals of Software Engineering (FSEN) (April 2011) (to appear)
21. Hildebrandt, T., Mukkamala, R.R., Slaats, T.: Safe Distribution of Declarative Processes. In: Barthe, G., Pardo, A., Schneider, G. (eds.) SEFM 2011. LNCS, vol. 7041, pp. 237–252. Springer, Heidelberg (2011)
22. International symposium on foundations of health information engineering and systems (August 2011)
23. Rahmanzadeh, A., Fox, J., Johns, N.: Disseminating medical knowledge: the proforma approach. Artificial Intelligence in Medicine 14, 157–182 (1998)
24. Khalaf, R., Leymann, F.: Role-based decomposition of business processes using BPEL. In: International Conference on Web Services, ICWS 2006, pp. 770–780 (September 2006)
25. Kindler, E., Martens, A., Reisig, W.: Inter-operability of Workflow Applications: Local Criteria for Global Soundness. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.) BPM 2000. LNCS, vol. 1806, pp. 235–253. Springer, Heidelberg (2000)
26. Lyng, K.M., Hildebrandt, T., Mukkamala, R.R.: From paper based clinical practice guidelines to declarative workflow management. In: Proceedings of 2nd International Workshop on Process-oriented Information Systems in Healthcare (ProHealth 2008), pp. 36–43, Milan, Italy (2008); BPM 2008 Workshops
27. Martens, A.: Analyzing Web Service Based Business Processes. In: Cerioli, M. (ed.) FASE 2005. LNCS, vol. 3442, pp. 19–33. Springer, Heidelberg (2005)
28. Milosevic, Z., Sadiq, S., Orlowska, M.: Towards a Methodology for Deriving Contract-Compliant Business Processes. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 395–400. Springer, Heidelberg (2006)
29. Mitra, S., Kumar, R., Basu, S.: Optimum decentralized choreography for web services composition. In: Proceedings of the IEEE International Conference on Services Computing, vol. 2 (2008)
30. Montali, M.: Specification and Verification of Declarative Open Interaction Models: A Logic-Based Approach. LNBIP, vol. 56. Springer (2010)
31. Mukkamala, R.R., Hildebrandt, T., Tøth, J.B.: The resultmaker online consultant: From declarative workflow management in practice to LTL. In: Proceeding of DDBP (2008)
32. Mukund, M.: From global specifications to distributed implementations. In: Synthesis and Control of Discrete Event Systems. Springer (2002)
33. Mulyar, N., Pesic, M., van der Aalst, W.M.P., Peleg, M.: Declarative and Procedural Approaches for Modelling Clinical Guidelines: Addressing Flexibility Issues. In: ter Hofstede, A.H.M., Benatallah, B., Paik, H.-Y. (eds.) BPM Workshops 2007. LNCS, vol. 4928, pp. 335–346. Springer, Heidelberg (2008)
34. Nanda, M.G., Chandra, S., Sarkar, V.: Decentralizing execution of composite web services. SIGPLAN Not. 39, 170–187 (2004)
35. Object Management Group BPMN Technical Committee. Business Process Model and Notation, version 2.0 (January 2011), http://www.omg.org/spec/BPMN/2.0/PDF
36. Rinderle, S., Wombacher, A., Reichert, M.: Evolution of Process Choreographies in DYCHOR. In: Meersman, R., Tari, Z. (eds.) OTM 2006. LNCS, vol. 4275, pp. 273–290. Springer, Heidelberg (2006)
37. Robertson, D.: A Lightweight Coordination Calculus for Agent Systems. In: Leite, J., Omicini, A., Torroni, P., Yolum, p. (eds.) DALT 2004. LNCS (LNAI), vol. 3476, pp. 183–197. Springer, Heidelberg (2005)

38. Sadiq, W., Sadiq, S., Schulz, K.: Model driven distribution of collaborative business processes. In: IEEE International Conference on Services Computing, SCC 2006, pp. 281–284 (September 2006)
39. van Glabbeek, R., Stork, D.: Query Nets: Interacting Workflow Modules That Ensure Global Termination. In: van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M. (eds.) BPM 2003. LNCS, vol. 2678, pp. 184–199. Springer, Heidelberg (2003)
40. Terenziani, P., Montani, S., Bottrighi, A., Torchio, M., Molino, G., Correndo, G.: The glare approach to clinical guideline: Main features. In: Symposium on Computerized Guidelines and Protocols, vol. 101, pp. 62–66 (April 2004)
41. van der Aalst, W.M.P.: Interorganizational workflows: An approach based on message sequence charts and petri nets. Systems Analysis - Modelling - Simulation 34(3), 335–367 (1999)
42. van der Aalst, W.M.P., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: Multiparty Contracts: Agreeing and Implementing Interorganizational Processes. The Computer Journal 53(1), 90–106 (2010)
43. van der Aalst, W.M.P., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. Computer Science - R&D 23(2), 99–113 (2009)
44. van der Aalst, W.M.P., Pesic, M.: A declarative approach for flexible business processes management. In: Proceedings DPM 2006. LNCS, Springer, Heidelberg (2006)
45. van der Aalst, W.M.P., Pesic, M.: DecSerFlow: Towards a Truly Declarative Service Flow Language. In: Bravetti, M., Núñez, M., Zavattaro, G. (eds.) WS-FM 2006. LNCS, vol. 4184, pp. 1–23. Springer, Heidelberg (2006)
46. van der Aalst, W.M.P.: Inheritance of interorganizational workflows: How to agree to disagree without loosing control? Information Technology and Management 4, 345–389 (2003)
47. Winskel, G.: Event Structures. In: Brauer, W., Reisig, W., Rozenberg, G. (eds.) APN 1986. LNCS, vol. 255, pp. 325–392. Springer, Heidelberg (1987)
48. Wodtke, D., Weikum, G.: A Formal Foundation for Distributed Workflow Execution Based on State Charts. In: Proceedings of the 6th International Conference on Database Theory, pp. 230–246. Springer, London (1997)
49. Yi, X., Kochut, K.J.: Process composition of web services with complex conversation protocols. In: Design, Analysis, and Simulation of Distributed Systems Symposium at Adavanced Simulation Technology (2004)
50. Zielonka, W.: Notes on finite asynchronous automata. Informatique Thorique et Applications 21(2), 99–135 (1987)

Towards a Formal Integrated Model of Collaborative Healthcare Workflows

Cristiano Bertolini[1], Martin Schäf[1], and Volker Stolz[1,2]

[1] UNU-IIST, Macau
[2] Dept. of Informatics, University of Oslo, Norway

Abstract. Health information systems (HIS) are becoming increasingly integrated through network communication technologies. Collaborative healthcare workflows (CHWF) are inherently complex, involving interactions among human actors, and (legacy) digital and physical systems. They are mission safety critical, privacy sensitive, and open to changes of requirements and environments. The complexity makes the definition, understanding, analysis, management, and monitoring of CHWF a software engineering challenge. We propose an approach to formal modeling and analysis of CHWF. The main problems that the approach addresses are *abstraction* and *separation of concerns* through algebraic manipulation. We use the CSP process algebra for modeling and verifying the dynamic interaction behavior of processes, and discuss the relation between the dynamic model and the static model of healthcare cases and resources. We use UML models to visualize the system's behavior and structure, but definitions of the syntax and semantics of these graphical models and their relation to the CSP models are left for future work.

Keywords: Integrated health information systems, collaborative workflows, infrastructure model, CSP, formal verification.

1 Introduction

The advances of computing and communication network technology offer a great potential to transform the public health service and reshape the future of health services globally. Governments and international organizations now have seen the opportunity to solve the pressing problems of constantly growing demand with limited resources in providing their peoples with safer, more effective, more patient centered, more timely, more efficient and more equitable health systems. A large number of national and international projects with huge amounts of funding [5, 13, 15, 21] are tackling this challenge. The common objective of these projects is to design and implement *integrated health information systems* (IHIS) through communication networks that provide effective support to *secure sharing of information and resources* across different health care settings and *collaborative healthcare workflows* (CHWF) among different care providers.

It is important, however, for all stakeholders of these projects to understand that there are many fundamental engineering challenges [12, 17] to successfully achieve this objective. The challenges are essentially due to the complexity of the system; the sources of the complexity include:

1. the interaction among a large number of different systems (clinic HIS, laboratory HIS, hospital HIS, pharmacy HIS, business management systems, etc.), with different models of data, access control and patient identification,
2. interactions of human actors (physicians, nurses, patients, etc.) with software systems and facilities,
3. conformance to conflicting and changing business rules and policies of different organizations, different ethical practices of different professionals, changing government regulations and laws, and
4. requirements on safety, security and privacy assurance.

Finding solutions for these problems calls for the application of systematic and rigorous software engineering techniques of modeling, design and validation that provide the means of abstraction, decomposition, separation of concerns and scalability.

An important method to tackle these problems is to separate the management of the coordination of these activities of collaboration using interconnected HIS in the realization of healthcare services. This technique is known as healthcare workflow management. Healthcare workflows involve interactions among different systems and stakeholders. In this paper, we look at the complex issues of CHWF and discuss the need for formal modeling and analysis. We focus on two issues of modeling and analyzing workflows: (i) how to describe workflows in a way to easily communicate with domain experts, and (ii) how to formally describe and verify them precisely.

We propose an integrated model of CHWF that supports the separation of the descriptions of different perspectives of CHWF: the resource model of the healthcare system organization in UML diagrams; the healthcare service case model that defines the static structure of the services, including the data functionality of the activities and the resources that are involved in performing the activities of the services. We show the dynamic behavior model of the processes in execution of the services using the process algebra CSP [6]. We also use UML activity diagrams for visual representation of the dynamic semantics of the CSP processes.

Organization. In Section 2, we briefly introduce the notion of workflows in general and discuss the desirable features and modeling elements of complex workflows, such as collaborative healthcare workflows, to motivate our methodology. As further motivation and related work, Section 3 gives an account of two major approaches to workflow modeling and discusses their limitations. The main contribution is in Section 4, a proposal for an integrated model of workflows. In Section 5, we illustrate our approach with a case study of a blood transfusion process. We draw in Section 6 our conclusions and discuss further challenges.

2 Information Systems for Healthcare Processes

In a business or a manufacturing organization, the process of providing a service or making a product involves the atomic activities of carrying out some *tasks* or *work items*. Each work item is carried by a unit of *resource*, which includes a member of staff or a piece of facility/equipment. *Workflow* or *business process management* is about coordination and control of activities in service processes. Workflow management ensures that the *flow of work items* of a service to be carried out are executed in an appropriate order by the right resource at the right time, so as to deliver the required service efficiently and effectively.

In early times, information systems, such as simple office information systems, were not networked and they were designed to support the execution of individual *work items*, such as filling in a form, in the process of providing a business service, such as a credit request to a bank. However, the workflow management was manual, because there is little to coordinate or control in the activities of a single user. When information systems became multiple user systems, especially when networks of systems emerged in the 1990s, software support to workflow management became possible, and also became required by business organizations. The Workflow Management Coalition defines a workflow management system as [18,22]:

> "A system that completely defines, manages, and executes workflows through the execution of software whose order of execution is driven by a computer representation of the workflow logic."

With the advance of computing and communication technologies, workflow management systems are becoming increasingly complex as well as increasingly crucial to the business development for organizations with geographically distributed offices, and for collaboration among different business organizations.

Our primary interest is in collaborative healthcare workflows among different healthcare institutions for secure, efficient and effective sharing of health information and care resources. Health information systems (HIS) used in health organizations support the execution of tasks, such as preparing an order, in healthcare services, such as a blood transfusion. The resources include patients, healthcare professionals, medical equipment/facilities and healthcare application software systems. The healthcare workflows must be managed to provide care services as safely, efficiently and effectively as possible.

A workflow management system is responsible for the tasks to be performed by means of the right resources (including people, facilities/equipment and application software systems) in the right order with the information they need, according to the business logic, but it does not actually perform any of the tasks in delivering a services. This is the separation of the *management* from the *applications* [19]. Therefore, the complete model of a workflow requires the definitions of the following models of different views:

– a **resource model** including its classification in terms of *roles* and the capability of the roles in terms of tasks that they can perform (activities);

- a **case model** in terms of the attributes of cases, tasks that can be performed in the cases and what changes to the attributes can be made when the task is performed, dependency relations among cases such as one case uses another, these attributes may be related to properties and states of domain objects;
- a **data model** defining the data and objects, including control data for workflow management, messages and information objects that the business processes communicate and change when tasks are performed;
- a **process model** specifying dynamic behavior of individual resources, and their collective behavior in workflows that defines the order in which tasks are performed according to the conditions of the case and the business logic.

The resource model also includes sub-information systems as legacy application software components in the network. This, thus, requires the resource model to include the black-box models of interface behavior of application software components. Each of the models is a hierarchical composition of sub-models of the same kind, and can be specified at different levels of abstraction. A methodology of modeling is required to reason about the consistency of these models to ensure the dynamic behavior of the process model and to analyze properties of the process behavior.

Systematic and precise definition of the models of the different views and their relations requires modeling notations. The optimal overall purpose of these models is the correct design of workflow management according to the business requirements. This is achieved through the use of models for the following objectives:

1. better understanding of the architecture and the behavior of the workflows,
2. validation of the correctness of definition of workflows with respect to intentions or requirements of the domain users and experts, through simulation,
3. formulation and verification of desirable properties of the workflows defined by the models,
4. analysis of feasibility and performance of the workflows by logic reasoning, verification and simulation, and
5. incremental refinement, decomposition/composition and separation of the design concerns of the different views of the workflow management.

These usages can only be effective and scale up when tools are developed to support simulation, analysis and verification. These tools can also be integrated as parts of the workflow management systems. The development of these tools requires the modeling notations to be formally defined, both syntactically and semantically.

Recall that our interest is in collaborative workflows of networked HIS of multiple healthcare organizations. The HIS are usually large-scale legacy systems with different models of data and access control. More seriously, much of the business logic has been hard coded into the application software of these legacy systems, and even many of its aspects might be incorrectly programmed. These organizations have competing business interests and adopt different policies and practice guidelines. Unlike other businesses management systems that mainly deal with production and administrative workflows [18] consisting of well structured and automated tasks within single organizations, collaborative workflows involve communication between processes, and between resources and non-automatic tasks. Modeling and analysis of these workflows are important for providing patients with safe and secure healthcare services, efficiently and effectively. This is yet a challenge, too.

3 Overview of Workflow Modeling

Before we propose our methodology, we give an account of the popular approaches to workflow modeling. We mainly consider Petri net models and UML-based models of workflows.

3.1 Petri Net Models

The most representative formal notation used for workflow modeling is Petri nets [8,19]. Petri nets is a notation for process dynamic behavior and thus naturally used for describing workflow processes. Petri net models can be visualized in a graphical notation, and this is, compared to other well-established formalisms, the unique advantage feature often claimed by users of Petri nets in workflow modeling. To us, the advantage of Petri nets is the explicit synchronization and true concurrency, plus the mechanisms of sequencing and choice naturally characterize the routing of tasks of workflows. Petri net models have a formally defined semantics in terms of transition systems, and thus formal verification is in principle possible. Though they may help intuitive understanding and communication of simple processes to non-computer scientists, the comprehension of graphs is always difficult to scale up, like all graphical notations. However, tools can be developed to allow graphical Petri net models to be constructed, and their semantic models automatically generated for verification. This to some extent eases the difficulty in building models with intuitive understanding. However, Petri net models of workflows have the following inherent disadvantages and limitations.

1. They do not support seamless integration with the model of the structure of the organization because Petri net models lose track of the relations between resources and their functionalities.
2. The cases are modeled as tokens of places that are also used as enabling conditions of transitions. Tokens do not distinguish the copies of processes of the same type and processes of different types. More importantly, representing cases this way makes it difficult to build a model of cases in which cases are place holders of their attributes, and to relate the dynamic behavior to the static relation between cases.
3. Petri net models have difficulties to represent data flows, i.e., communication of data between processes. It is also difficult to model conditions that relate to attributes and information objects. These are required for modeling collaborative workflows that are typical for integrated HIS for effective sharing of health

information and care resources. Indeed, Petri net models are extended with colors for these, but this complicates understanding even further.

4. Extension of Petri nets to model real-time workflows is complicated. Mechanisms of exception handling and compensation can be modeled only by concrete use of atomic activities and routing mechanisms.
5. Finally, but possibly most importantly in terms of features of formal notation, the notation of Petri nets is not an architecture description notation. Thus, it does not support compositional reasoning and verification. Furthermore, it does not support algebraic reasoning and derivation of workflows that are often significant for understanding and judgment of correctness of workflows, which require semantic based verification.

3.2 UML-Based Models

The Unified Modeling Language (UML) [1] is a standardized general-purpose modeling language for object-oriented software systems. It includes a set of graphical notations and corresponding techniques for the creation of visual models of different perspectives (or views) of object-oriented software systems. UML diagrams are used to represent two different views of a system model:

– *Static* (or *structural*) including class diagrams, object diagrams, component diagrams and use-case diagrams views, which represent aspects of the static structure of a system.
– *Dynamic* (or *behavioral*) views including sequence diagrams, activity diagrams and state machine diagrams (statecharts), which emphasize the dynamic behavior of the system by showing interactions among objects, and changes of system and object states.

The popularity of UML is due to ease of communication among customers, designers and developers, and its standardization that the software industry and tool development need. Indeed, UML has contributed greatly to improvements in the development of CASE tools.

UML is also proposed for business process modeling [7], and a standardization effort is being made for UML models of workflows [14]. The commonality of these UML-based approaches, as surveyed in [20], is that they all focus on describing the dynamic behavior of workflows by using UML behavioral models, sequence diagrams, activity diagrams and statecharts, leaving the model of resources, attributes of cases and data imprecise.

We recognize the contribution of UML to separation of concerns by proposing that different views and aspects of the model of a system be modeled by different notations. Also, tool support to create graphical models and automated processing and transformation help to ease the difficulty in model building. However, multi-view modeling has also created the serious difficulty of ensuring the semantic consistency of the models. This is even an unsolvable problem for UML,

[1] `http://www.omg.org/spec/UML/2.0/`

because it is a general modeling language and it does not have a formally defined semantics. Also, UML diagrams alone do not form the full model of a system, and textual documents, such as written use cases and comments on diagrams, are contained in the model. Therefore, tools are mostly concerned with syntactic/diagrammatic manipulation and consistency checking.

Also because of the lack of formal semantics, UML does not directly support formal reasoning about relations between models and analysis. For example, dynamic behavior of workflows is described at different levels of abstraction by use-case diagrams, sequence diagrams, activity diagrams and state machine diagrams [20], but there are no definitions of the semantic relations and refinement relations between these diagrams.

There is little literature focusing on healthcare workflow modeling. Recently, a graphic notation, called Little-JIL, has been defined with an interpretation to finite state machines. It is used for simulation and verification of medical processes [2]. It describes a medical process in a top-down manner similar to recursive procedure definition, e.g., a graph representation of $P = P_1;\ P_2$, or $P_1 = P_{11} \mid\mid P_{12}$, etc. Pre- and post-conditions can be associated with subprocesses. However, it shares the major disadvantages that Petri nets and UML diagrams have. Little-JIL is also used to identify exceptions and describe exception handling, but at a level similar to directly programming the error detection and handling of errors.

3.3 Survey Conclusion

Though workflow management technologies, such as COSA, IBM Flowmark and Microsoft Exchange, are available, and standards such as those of the Workflow Management Coalition and the OMG [14,16], exist for workflows, and workflow modeling, formal workflow modeling in particular is still in its infancy. The application of workflow management in healthcare systems also lags behind other business applications, such as banking, insurance, or stock management. Collaborative integration of workflow management of multiple and heterogeneous systems is a phenomenal problem in healthcare systems, and modeling such workflows is a fundamental challenge.

4 Proposed Methodology of Workflow Modeling

We propose an integrated model of workflows to support design, validation, verification and analysis of workflow management systems. It aims to support separation of concerns. The proposal is heavily influenced by the rCOS method of use-case driven design of object-oriented and component-based application software systems [3], and the separation of business process management from the functionality of the application software.

4.1 rCOS Overview

Use-case driven design [9] is a top-down process in which software requirements at the top-level are identification and description of models of use cases. The use case model consists of the actors, the business processes they are involved in, and the relations between use cases, represented by UML *use case diagrams*. The use case model also contains a document of use cases that describes the actions (or *activities*) of using the system (or sending *events* to the system) in performing operations (or invoking methods) on domain data and objects (i.e. changes of states of the software), and the pre- and post-conditions of the operations. The order in which activities are performed is also described or represented as UML *sequence diagrams* according to the business logic. The use case descriptions also identify the conceptual domain objects, their properties (data attributes) and relations. These objects and relations are classified and organized in a *class model* represented by a UML class diagram. In the rCOS method, the use-case model and the conceptual class model together are called the *requirements model* [10].

Use-case diagrams are used for the purpose of informal communication and understanding. Their formal counter parts are *component diagrams*. Each use case has

1. a set of data attributes that is assumed to contain a single variable x for simplicity of the discussion;
2. a provided interface that declares a set of methods $m(in, out)$ that can have input and return parameters: these are the events that the actors of the use case send to the system in the use case sequence diagram;
3. a functional specification of each method $m()\{p(x, in) \vdash R(x, in, out', x')\}$ meaning that if the state x and the input in satisfies the *precondition* p, the execution of the method terminates in a state x' with a return out' such that the post-condition $R(x, in, out', x')$ holds.

The types of the attributes can be classes or pure data types, such as integers, Boolean values and characters. A class diagram that is formalized in a type system defines these types. A simple sequence diagram, representing the interaction of the actors with the component, can represent the protocol. The rCOS tool includes a translator of sequence diagrams to CSP processes.

Open components that have required interfaces, as well as provided interfaces, are defined. This allows defining composition of components, plugging, parallel composition and coordination of components to model relations between use cases by UML component-diagrams. Therefore, the requirements model of a system has a component-based architecture:

1. the use case model becomes a component diagram for the static structure, the pre- and post-conditions for the data functionality, and the protocols for dynamic behavior; and
2. the data types are defined by a class diagram.

This architecture model has a formally defined semantics.

The design activity is to design each interface method in each component (that models a use case) and assign the functional responsibility specified by the pre- and post-conditions of the method to methods of the objects of the attributes of the component. This involves decomposition and refinement of the pre- and post-condition specification, and the practice of the Design Patterns of Responsibility Assignments in OO design [9]. Graphically, the decomposition and assignment of responsibility transform the sequence diagram to an object-sequence diagram, describing the implementation of the use-case operations by interactions among objects. Also, the classes in the conceptual class model are refined to design classes (or software classes), with the responsibilities assigned to their objects being represented as their methods. Thus, an OO design model is obtained with the object sequence diagrams and the design classes.

To obtain a component-based design, we take each object sequence diagram and identify the appropriate objects (that must be permanent) and make them *components*. The other objects are then properly "wrapped" as fields of these "component objects" and the object-sequence diagram is transformed to a component-sequence diagram. The rCOS tool aids in transforming the object-sequence diagram into a component sequence diagram and generates a component-diagram, once the component identification is done by the tool user.

The two steps refine a component corresponding to a use case into a composition of components. After all use-case components are defined this way; the component-based model of the requirements is refined to a component-based design model.

4.2 Component-Based Model of Workflows

We now extend the rCOS method to workflow modeling. The key idea to solve the third and fourth problems in use-case driven software design is to detach the interaction between actors of a component from the data functionality. This can be easily done by modeling a use case purely as an event handling component, called a *use-case handler*, that only takes method invocations from actors and delegates them to the software component that implements the functionality, called an *application software component*. To do this, we first extract from the attributes of a use-case component defined in the previous subsection the variables that are used only for the control of protocol. Such an event handler can be defined as an rCOS open component to be plugged to the application component. However, the detachment allows, by using middleware components or specially design connectors and coordinators, even looser coupling interactions with the application component than method invocations and more flexibility in interaction and synchronization with other use case handlers. These are also requirements for modeling collaborative workflows. With this discussion, we define our framework of workflow modeling in terms of a *case model* and a *process model*.

Case Model. This is similar to a use-case diagram that contains use cases, the resources that are involved in each case, and relations between cases. However, this model is organized as a diagram in which

– A case class corresponds to a use case of a use-case model, and it is annotated with a stereotype ≪*case*≫ and shows its attributes, events and their functionality specification. A case model has a designated *top case*.

- A resource class corresponds to an actor in a use case model. It is annotated with a stereotype ≪*resource*≫ and its attributes and events in the method section as its capability. In the following discussion we focus on the events, ignoring attributes and functionality on attributes.
- The association from a resource class to a process call is labeled by the set A of events that the resource performs in the process, that is the interaction interface between the resource and the process.
- An association "◇" from one case, called a *choice composite case*, to a number of cases, called the *choice component cases* of the composite case, represents external choice. An association "⧫" from a case, called a *parallel composite case* to a number of cases, called the *parallel components*, represent a parallel composition (PCV in Fig. 1). An association between one component case and another is labeled by a set A of events to represent their synchronization events (see in Fig. 1 the *wait*-association between cases $Admission$ and $Wonder$ and $d_1 \dots d_n$-association between case $Diagnosis$ and case $Treat$), i.e. interfaces of interactions between the two cases. If the set is empty the association is not shown in the diagram and the two component cases are independent.

Therefore, a case model is an extension to a use case model that is more informative, and precisely defines the interfaces between cases and resources. This is important for collaborative workflow modeling, because

- interface events model how events (or actions) are jointly performed by different resources, including human resources and application software components (nature of agent-orientation),
- the resources can be distributed among a network of organizations (systems),
- the sub cases in the case model can be cases from different organizations using or supported by different application software systems, and
- what events are automated by software and what events are not automated though being monitored and coordinated.

The use of interface events between parallel compositions of cases is essential for component-based workflow modeling. Furthermore, structural relations between use cases, such as components of parallel compositions and choices, are to be formally defined as CSP processes. Therefore, such a model does not have the problem of the semantically undefined relations of "includes" and "extends" between use cases. In practice, cases do not usually have recursive structures and the case model is a tree structure. However, in theory the process of a component case C_1 of a parallel composite or a choice composite case C can call for the process of composite case C to repeat. Therefore, a case model in general is a directed graph structure.

Event-based interactions among cases are the most significant concepts of collaborative and concurrent processes, and sequencing relation between processes of cases are often realized by synchronization events, i.e. triggering events. Just as the philosophy of observing interactions of communicating processes in CSP [6] and CCS [11], sequential composition in collaborative workflows modeling is a second class concept and can be defined by synchronization, and it is not explicitly modeled in a case model.

It is important to note the difference between the notions of cases and work items (or activities), and those of business processes of cases and events (or activities) that execute work items. A business process represents the story of performance of events for a case from the start and the end is providing a service or making a production, while an event is the performance of a work item or an activity. Thus activities are atomic. It is important to note that a business process can be represented as a CSP process.

A case model is a static structural model of cases for a top-level (composite) case. It is a *type model*, that is, a classifier model in UML. An instance of it can be represented as a UML object diagram, in which resources are instantiated to individual staff members and pieces of facilities. Consider the case of *patient clinic visit*, denoted by *PCV*. It involves resources of *patient*, *admission nurse*, *nurse*, *doctor* and *clinic HIS*.

The events of a case, e.g. *Admission*, can be *verifyPatient* and *checkBlood-Pressure* jointly carried out by the *patient*, the *admission nurse* and the *clinic HIS*. The parallel composition in *Wait* is for illustration purposes. The set of events for the labels that associate a resource to a case is given by the events that the resource participates in the process.

A Brief CSP Overview. We will propose two related approaches to modeling dynamic behavior or processes of cases, *case-oriented* and *resource-oriented* modeling. They both use the well-established process algebra CSP [6].

We use CSP [6] to define the behavior of a case as a process expression written in the syntax defined in equation 1.

$$P ::= a \mid P_1;\ P_2 \mid P_1 \sqcap P_2 \mid P_1 \Box P_2 \mid P_1 \parallel P_2 \mid P \backslash A \mid \mu X.P(X) \\ \mathbf{SKIP} \mid \mathbf{STOP} \tag{1}$$

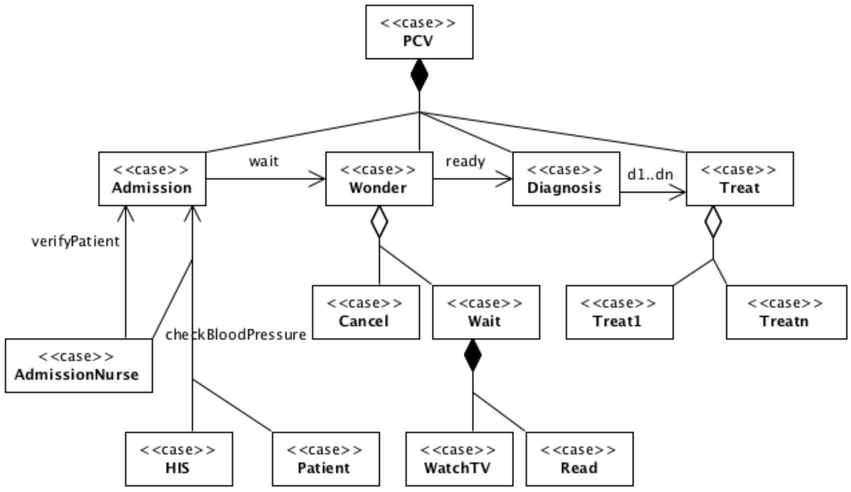


Fig. 1. Case Model Example

where a is an event, $P_1;\ P_2$ is sequential composition, $P_1 \;\Box\; P_2$ is external choice, $P_1 \sqcap P_2$ and $P \backslash A$ are internal choice and hiding as abstraction mechanisms, $P[b/a]$ renames the event a for reuse of processes, $P_1 \parallel P_2$ is parallel composition, **SKIP** and **STOP** are respectively the process that does nothing but terminate and the process that deadlocks immediately. $\mu X.P(X)$ denotes recursive equation definitions, and it can be written as the equation $X = P(X)$, where X is a process variable.

When attributes become significant and value passing between processes is needed, events can send a value $c!(v)$ and receive a value $c?(x)$. Guards of processes can also be specified as $g\&P$ meaning when g holds on the current event the P can be executed and it deadlocks otherwise. The internal choice and hiding are abstraction mechanisms for information hiding for verification of properties. Note that the case model does not use them.

For CSP, there are three complementary semantics models, *trace model*, *stable failure model* and *failure-divergence model* for verification of safety, deadlock freedom and divergence freedom properties of interactions among processes. A refinement calculus of CSP expressions is developed for each of these semantic models to support algebraic reasoning. Verification of properties and refinement are supported by the model checker FDR2 [2].

Case-Oriented Process Model. Each case C is assigned a CSP process $\underline{C}$ whose alphabet is the set of the events of the case. Then, $\underline{C} = \underline{C}_1 \;\Box\; \ldots \;\Box\; \underline{C}_n$ if C is a $\Diamond$-aggregation of $C_1, \ldots, C_n$, and $\blacklozenge$-aggregation is defined to be the parallel composition of the components, synchronized on the sets of events that label the associations between the components of the composition. The process of each case in the case model is then specified as a CSP process, and the whole case model is defined as a CSP process. For example, the process model of the case model of PCV can be defined as

$$\begin{array}{ll}
PCV & = Admission \parallel Wonder \parallel Diagnosis \parallel Treat \\
Admission & = verifyPatient;\ (checkBloodPressure;\ wait) \;\Box\; nOk \\
Wonder & = wait;\ Cancel \;\Box\; Wait \\
Wait & = Read \,||\, watchTV \\
Read & = (read;\ Read) \;\Box\; ready \\
watchTV & = (watch;\ watchTV) \;\Box\; ready \\
Diagnosis & = ready;\ d_1 \;\Box\; d_2 \\
Treat & = (d_1;\ Treat_1) \;\Box\; (d_2;\ Treat_2)
\end{array} \qquad (2)$$

This approach is called *case-oriented* because the CSP processes identify and represent the events as whole actions jointly performed on work items of the cases. There is no explicit information about the contributions of the resources to the events. This approach focuses on composition of services and coordination of service activities. It is very much the same as service-oriented modeling.

[2] `http://www.fsel.com/`

Resource Oriented Modeling. The idea of *agent-oriented* modeling is followed in this approach. We identify the behavior that each resource R performs in the process of a case C and represent it as a CSP process R-in-C, and then parallel compose the processes of all the resources in the case, synchronized on the same activities, i.e. events that are contained in more than one resource. The CSP process for a case C is then defined as the following form

$$R_1\text{-in-}C \parallel R_k\text{-in-}C$$

And each R_i-in-C has a structure of determined by the structure of the case C defined in the case model. For the example case PCV, the process of admission nurse in PCV is defined as

$$AdmissionNurse = (verifyPatient;\ checkBloodPressure;\ wait) \,\Box\, nOK$$

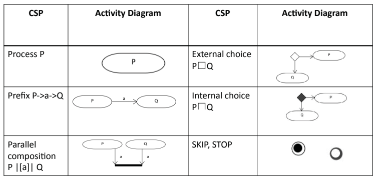
Note that we omit the event nOK in the example case and in Figure 1 because it represents an exception in the workflow. It can be modeled as another CSP process. However, we do not deal with exceptions in this small example.

The consistency of a case-oriented process model and a resource-oriented process model for a case model is defined, and reasoning about equality is supported through the algebraic laws of CSP processes.

Advantages of the Methodology. In addition to the power of CSP, the proposed methodology enjoys the following advantages:

1. It allows one to create a fully integrated model of workflows, yet supports separation of concerns.
2. It combines both techniques of service- and agent-oriented modeling.
3. Techniques and models of the component-based design of the application software systems of the organizations can be reused in workflow modeling.
4. The models support component-based system evolution. New workflows can be defined and the underlying application components can be extended to support the newly added workflows.
5. Consistency checking of workflows can be done by checking deadlock and livelock freedom of the CSP process obtained from the model through FDR.
6. Feasibility of workflows can be checked if the interfaces of resources and the cases are supported by the resource model.
7. The methodology can be easily extended to deal with real-time using real-time CSP, and to deal with exception handling and compensation using Compensating CSP (cCSP) [4].

A possible concern from customers and end users about the usability of CSP processes is the lack of support for their visualized representation. In fact, this is not a big issue as graphical modeling and tool support can be developed. We propose the use of UML activity diagrams, a combination of statecharts and Petri nets, for graphic representation of the dynamic behavior of CSP processes. However, the UML activity diagrams are extended with states for normal termination and deadlock, and internal choice. The translation between CSP processes and activity diagrams is shown in Fig. 2, and Fig. 3 presents the activity diagram of the PCV example. However, like all graphic notation for dynamic behavior of interaction processes, activity diagrams only depict the operational semantics and trace semantics of CSP. For specification and verification of general failures and divergence properties, the more sophisticated stable failure semantics and failure-divergence semantics are needed. As we observe, Fig. 1 represents the static structural model and Fig. 3 represents the dynamic model.



CSP	Activity Diagram	CSP	Activity Diagram
Process P	P	External choice P☐Q	P Q
Prefix P->a->Q	P a Q	Internal choice P⊓Q	P Q
Parallel composition P |[a]| Q	P Q a a	SKIP, STOP	◉ ○

Fig. 2. CSP and Activity Diagram map

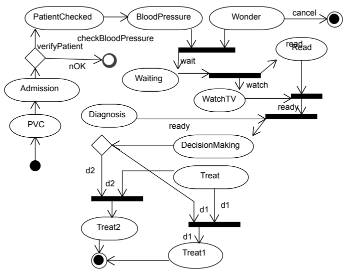


Fig. 3. Activity Diagram of PCV

5 Case Study: Blood Transfusion System

In this section, we present a blood transfusion system case study. We apply our previously defined methodology to modeling and analyzing such a system.

The case study is based on a blood transfusion system benchmark [1,2]. The system consists of a nurse, a physician and a patient. The nurse must collect the patient's consent and the physician's authorization and also checks the patient identification and material for the transfusion. The physician must fill out the transfusion order. The patient must provide his/her information and also agree with the blood transfusion procedure. The nurse is responsible to collect the information from the patient, his/her consent, and the order from the physician. The nurse also carries out the blood transfusion and monitors the patient to handle an eventual adverse reaction to the transfusion.

For the blood to be prepared, first the blood bank must check if it is available, and if the patient's blood type is known. In case the blood type is not known, the blood bank collects a blood specimen to find out the patient's blood type. During the blood transfusion some information must be collected from the patient. Before starting the transfusion the nurse must check all information about the patient, whether the type is known, and whether the physician's order is completed. The patient is monitored and in case of any reaction the procedure is stopped immediately. Finally, all information from the patient and the blood are checked again.

Case Model. Fig. 4 show the case model for the blood transfusion system. It presents the cases and resources, which are related with the cases by the

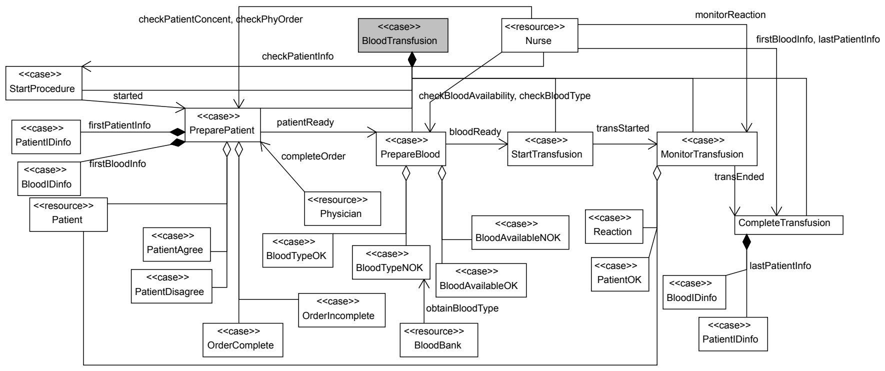


Fig. 4. Case Model

synchronized events according to the informal requirements described in the benchmark: before the blood transfusion, the patient and the blood must be prepared; the nurse must obtain a specimen from the patient; then the blood transfusion starts; and in parallel the procedure must be monitored and any transfusion reaction must be handled. The informal requirements are easily modeled in a case model.

The cases are composed by the main functional requirements. The main flow involves the cases *StartProcedure*, *PreparePatient*, *PrepareBlood*, *StartTransfusion*, *MonitorTransfusion* and *CompleteTransfusion*. The other cases are used to model choices and to synchronize with specific requirements of the main cases. The resources are composed by the human actors *Nurse* and *Physician*, and the system *BloodBank*. We obtain the following **process model** with the derived information about the composed processes, and the detailed specification in the form of CSP of the sub-cases:

$$\begin{array}{ll}
\mathit{BloodTransfusion} & = \mathit{StartProcedure} \parallel \mathit{PrepareProcedure} \parallel \mathit{PrepareBlood} \\
& \parallel \mathit{StartTransfusion} \parallel \mathit{MonitorTransfusion} \parallel \\
& \mathit{CompleteTransfusion} \\
\mathit{StartProcedure} & = \mathit{checkPatientInfo};\ \mathit{started} \\
\mathit{PreparePatient} & = \mathit{firstPatientInfo};\ \mathit{firstBloodInfo};\ \mathit{checkPatientConsent}; \\
& (\mathit{PatientAgree})[\,](\mathit{PatientDisagree});\ \mathit{checkPhyOrder}; \\
& (\mathit{OrderComplete})[\,]\mathit{OrderIncomplete});\ (\mathit{patientReady})[\,]\mathit{Skip}) \\
\mathit{PrepareBlood} & = (\mathit{BloodTypeOK}[\,](\mathit{BloodTypeNOK};\ \mathit{obtainBloodType})); \\
& (\mathit{BloodAvailableOK}[\,]\mathit{BloodAvailableNOK});\ \mathit{bloodReady} \\
\mathit{StartTransfusion} & = \mathit{transStarted}; \\
\mathit{MonitorTransfusion} & = \mathit{transStarted};\ (\mathit{Reaction}[\,]\mathit{PatientOK});\ \mathit{transEnded} \\
\mathit{CompleteTransfusion} & = \mathit{transEnded};\ \mathit{firstBloodInfo};\ \mathit{lastPatientInfo};\ \mathit{Skip} \\
\mathit{PatientIDinfo} & = \mathit{firstPatientInfo};\ \mathit{lastPatientInfo} \\
\mathit{BloodIDinfo} & = \mathit{firstBloodInfo};\ \mathit{lastBloodInfo} \\
\mathit{Nurse} & = \mathit{checkPatientInfo};\ \mathit{checkPatientConsent}; \\
& \mathit{checkPhyOrder};\ \mathit{checkBloodAvailability}; \\
& \mathit{checkBloodType};\ \mathit{monitorReaction} \\
\mathit{Physician} & = \mathit{completeOrder} \\
\mathit{BloodBank} & = \mathit{obtainBloodType}
\end{array}$$

Workflow analysis After having formally described our case study as a set of CSP processes, we now give a property for which we can achieve verification using algebraic manipulation on the process defined by the graphical model.

For example, let's consider the following property describing the blood transfusion [1]: *P1: before infusing each single unit of blood product into a Patient, that Patient's ID information must be checked.* It means that the patient ID information must be verified before the patient starts the transfusion. In CSP we can say that the event `\verifyPatient` must be followed by the event `\startTransfusion`. It is easy to verify this property in CSP because we can simply hide all events that are not part of the context. Then we check $SYSTEM \sqsubseteq P1$. Other properties that the CSP hiding operator can be used for are related to the human and non-human actions. In this way, we can verify only actions related with the implemented system or only actions related with the human interaction. For example, the nurse has to check the information is complete in the patient's ID band or if the patient is feeling well (there is no action in the system but it is human action). Then we can verify only the system's actions, just hiding the human's action.

6 Conclusions and Future Work

This paper has first discussed the need to design advanced workflow management systems to support the integration of healthcare systems through the new computing and communication network technology. We argued that for the design of such workflow management systems, the current state of the art of workflow modeling, validation, verification and analysis needs to be improved. The main contribution is a proposal of modeling methodology that supports the reuse of model-driven and component-based techniques and tools in application software modeling and design for workflow modeling. The main advantage of the proposed methodology is the semantic integration of models of separated views of workflow models, the static views of the organization, the cases and the processes view of the workflows. The proposal has been based on the advantages of the event-based process algebra CSP. It is used in this paper, over the other formalisms that are currently used for workflow modeling. Algebraic reasoning allows the consistent combination of both service-oriented and agent-oriented techniques. The ability of CSP to describe architectural operations is also important for component-based modeling. Yet, the abstraction mechanism of hiding and internal choice for information hiding allows the specification and refinement of architecture models at different levels of abstraction.

The use of black-box interface models for application components provides clear separation of and interfaces between workflows and application software components, providing flexibility of system evolution. Application software components become resources of workflows in our methodology and clearly identify automated and non-automated activities. The use case model then represents the collaborative nature of resources in processes of services. The methodology is illustrated with a motivating example and a benchmark case study.

Future Work

This paper has largely focused on conceptual discussion. Most of the formal technical details have yet to be worked out. However, we do not see insurmountable challenges there. Further fundamental challenges that we foresee include the following problems.

1. There is a need to test the methodology on a real benchmark case study with the advanced issue of collaborative healthcare workflows.
2. Workflows are closely related to policies, practice guidelines, and regulations. All workflow modeling approaches, including the one proposed in this paper, have these policies and guidelines hard coded into the temporal order of the events of the workflows. It would be very much desirable for policies, (such as privacy and security policies) to become first class citizens in the model, so that workflows can be designed and monitored in explicit accordance to these policies.
3. For networked organizations, processes of cases cannot be short-lived (atomic) transactions. They are rather Long Running Transactions (LRTs). Most business process languages, such as BPEL, provide abstract facilities for programming LRTs. It is thus important to extend the case model and the process model to permit the modeling of LRTs, or compensation when exceptions occur. For the process model, we will use the extension of CSP, called compensating CSP (cCSP).

Acknowledgements. Supported partly by the projects GAVES and ARV funded by the Macau Science and Technology Development Fund and NSFC-600970031.

References

1. Blood Transfusion Medical Benchmark, `https://collab.cs.umass.edu/groups/laser_library/wiki/04be2/BTBenchmarkDownloads.html`
2. Chen, B., Avrunin, G.S., Henneman, E.A., Clarke, L.A., Osterweil, L.J., Henneman, P.L.: Analyzing Medical Processes. In: Proceedings of the 30th Intl. Conf. on Software Engineering, ICSE 2008, pp. 623–632. ACM, New York (2008)
3. Chen, Z., Liu, Z., Ravn, A.P., Stolz, V., Zhan, N.: Refinement and verification in component-based model-driven design. Sci. Comp. Prog. 74(4), 168–196 (2009)
4. Chen, Z., Liu, Z., Wang, J.: Failure-Divergence Refinement of Compensating Communicating Processes. In: Butler, M., Schulte, W. (eds.) FM 2011. LNCS, vol. 6664, pp. 262–277. Springer, Heidelberg (2011)
5. eHealth Initiative. National progress report on ehealth (July 2010), `http://www.ehealthinitiative.org`
6. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall (1985)
7. Hruby, P.: Specification of workflow management systems with UML. In: Proc. of the OOPSLA 1996 Workshop on Business Object Design and Implementation (1998)
8. Jensen, K.: Coloured Petri Nets: Basic concepts, analysis methods and practical use. Springer, Berlin (1996)

9. Larman, C.: Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process, 2nd edn. Prentice-Hall (2001)
10. Liu, Z., He, J., Li, X., Chen, Y.: A Relational Model for Formal Object-Oriented Requirement Analysis in UML. In: Dong, J.S., Woodcock, J. (eds.) ICFEM 2003. LNCS, vol. 2885, pp. 641–664. Springer, Heidelberg (2003)
11. Milner, R.: Communication and Concurrency. Prentice Hall (1989)
12. National Academy of Enginnering. Grand challenges for engineering (2008), `http://www.engineeringchallenges.org/cms/challenges.aspx`
13. National E-Health Transition Authority. NEHTA Blueprint. Version 1.0, Draft for Consultation (August 2010)
14. OMG. UML extensions for workflow process definition - request for proposal. OMG-document bom/2000-12-11 (December 2000)
15. Mechael, P., et al.: Barriers and gaps affecting mHealth in low and middle income countries: Policy white paper. Technical report, Center for Global Health and Economic Development Earth Institute, Columbia University (May 2010)
16. Rational. Business modelling with the UML and Rational Suite AnalystStudio. A Rational Inc. Software White Paper (2000)
17. Shepherd, M.: Challenges in health informatics. In: Proceedings of the 40th Hawaii International Conference on System Sciences (2007)
18. van der Aalst, W.: The application of petri nets to workflow management. The Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)
19. van der Aalst, W., van Hee, K.: Workflow Management: Models, Methods, and Systems. MIT Press (2002)
20. van der Aalst, W., Weske, M., Wirtz, G.: Advanced topics in workflow management: Issues, requirements, and solutions. Journal of Integrated and Process Science 7(3), 49–77 (2003)
21. Vital Wave Consulting. mHealth for development: The opportunity of mobile technology for healthcare in the developing world. UN Foundation-Vodafone Foundation Partnership, Washington, D.C. (2009)
22. WFMC. Workflowmanagement coalition terminology and glossary (WFMC-TC-1011). Technical report, Workflow Management Coalition, Brussels (1996)

NOVA Workflow: A Workflow Management Tool Targeting Health Services Delivery

Wendy MacCaull and Fazle Rabbi

Centre for Logic and Information,
St. Francis Xavier University
{wmaccaul,rfazle}@stfx.ca

Abstract. We present the NOVA Workflow tool-suite, a prototype for a process, information and communication management tool to guide and inform real world workflows with special attention to the needs of health services delivery. NOVA Workflow is an innovative workflow management system which integrates formal verification into the software development process. For workflow modeling the tool uses the time Compensable Workflow Modeling Language (CWML_T) which produces reliable and structured workflow models and enhances error handling. The graphical editor of the tool gives a common platform for modeling, verifying and developing software. The SOA based architecture of the workflow engine ensures compliance with industry standards. The tool includes an automated translator to a model checking tool, a monitor to facilitate run-time compliance of (health care) policy, and a user friendly browser to give clinicians a convenient way to view a patient's information without losing the context. We propose an application of the browser to process diagnosis.

1 Introduction

This paper presents an integrated approach for modeling, verifying, developing and monitoring workflow management systems (WfMSs), with special attention to the needs of safety critical systems such as health care systems. A report estimated that approximately 98,000 deaths per year in the United States were the result of medical errors, many of which could be traced to faulty processes [18]. Errors not leading to death are costly and adversely affect the patient. WfMSs can help ensure compliance with protocols. Model checking processes in these systems, before enactment, can save time and reduce errors, while using a model checked monitor can alert clinicians to abnormal situations. However, commercial WfMSs do not have adequate rollback mechanisms (for error handling and side effects) and many model checkers deal only with relative (rather than quantified) time eg.: when an emergency case arrives at the hospital, standard model checking can only verify whether, for a particular process *"Eventually the patient receives the treatment"*, but to save the patient's life, it should be verified that *"The patient receives the treatment within half an hour"*.

In this paper we present an integrated tool for developing and verifying enterprise software systems. Rather than verifying actual programs, an abstract specification for the software is written and used to verify properties of the system. The abstract specification is written using a limited syntax in Java and the specification is translated to a model for a model checker. Enterprise software usually consists of hundreds if not thousand of components; each component has many business logics, data access operations and third party service invocations. In addition, client applications require an enormous programming effort to provide sophisticated Graphical User Interfaces (GUI). To verify such complex software systems without abstraction is challenging. Our tool-suite NOVA Workflow[1] deals with this problem by abstraction (i.e., abstract process specification) and reduction which makes it feasible to verify enterprise and/or safety critical software systems.

The NOVA Workflow tool suite has five components, i) the NOVA Editor, ii) the NOVA Translator, iii) the NOVA Engine iv) the NOVA Monitor and v) the NOVA Browser. The NOVA Editor uses the graphical modeling language, the Compensable Workflow Modeling Language, extended with the time constraints of delay and duration (CWML$_T$). The NOVA Translator translates the workflow model and Java specification into DVE, the modeling language for the parallel distributed model checker DiVinE [5]. The NOVA Engine is a workflow engine based on Service Oriented Architecture (SOA). The NOVA Engine can be used in a system as a workflow library and it does not provide any restriction on application development. The engine was developed on the Spring [10] and Hibernate [9] platforms both of which can be deployed to various application servers. Spring is a widely used open source framework that helps developers build high quality applications faster. Hibernate is an object-relational mapping (ORM) library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. We used a three tire architecture for NOVA Workflow as centralizing the business logic in an application tier has several advantages including maintainability, extensibility, and interoperability. The NOVA Monitor integrates time constrained monitors with workflow models. The NOVA Browser is a flexible user interface designed to allow brainstorming to enhance the user experience. The integrated tool support for modeling, verification and development of workflow management systems together with the monitor will greatly help its users build reliable safety critical systems. Fig. 1 shows the architecture of NOVA WorkFlow.

The rest of this paper is organized as follows. The components of the NOVA Workflow are described in section 2, (the NOVA Editor) section 3, (the NOVA Engine) section 4, (the NOVA Translator) section 5, (the NOVA Monitor) and section 6 (the NOVA Browser). Section 7 presents a case study and Section 8 discusses related work and concludes the paper. More details, case studies and proofs pertaining to the NOVA Editor, Translator, Engine and Monitor may be found in [30,27,29]. Most of the information on the NOVA Browser, including the proposed application, appears here in published form for the first time.

[1] `http://logic.stfx.ca/software/nova-workflow`

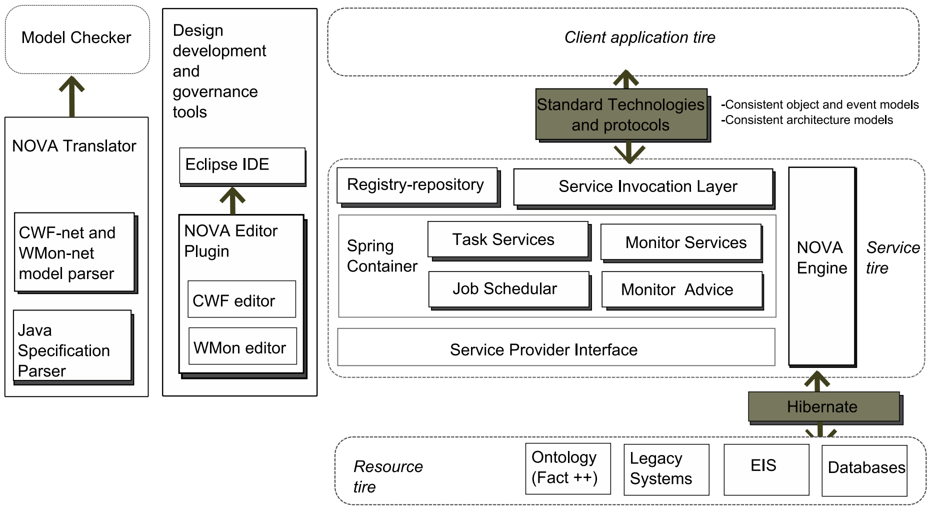


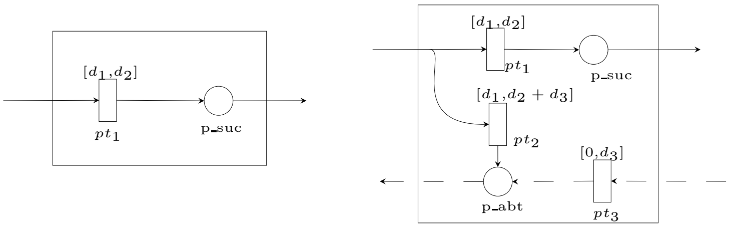
Fig. 1. SOA based architecture of NOVA workflow

2 The NOVA Editor

The NOVA Editor is a visual modeling tool for the time Compensable Workflow Modeling Language, CWML$_T$ [30,27]. A compensable workflow model consists of compensable and uncompensable tasks. In [30] we defined CWML (the untimed version of CWML$_T$). In addition to the basic operators "sequence" ($\bullet$), "and" ($\wedge$), "xor" ($\times$), "or" ($\vee$) and "loop" ($^+$), CWML uses the t-calculus operators [25] (sequential composition (;), parallel composition ($||$), internal choice ($\sqcap$), speculative choice ($\otimes$), and alternative choice ($\rightsquigarrow$)), to model compensation. In [27] we extended CWML with time, by including the notions of delay and duration and called it CWML$_T$. Timing constraints for most workflows can be expressed using delay and duration [26,19]. The foundations of CWML$_T$ are essentially time Petri nets (with integer valued time) referred to as Explicit Time Petri nets in [27]. Integers rather than reals suffice to model processes in health services delivery. A hybrid Petri net based semantics incorporating both weak and strong semantics is used [27] to model forward and compensation flows.

Atomic tasks in CWML$_T$ are of two types, uncompensable and compensable. An uncompensable atomic task is an activity which always finishes successfully, if activated. In case of an error executing the forward flow, a compensable task aborts and performs some compensation. The Petri net based representation of an atomic uncompensable task and an atomic compensable task with time constraints are given in Fig. 2. The Petri net representations of compound tasks may be found in [30,29].

In Fig. 2 solid arcs represent a forward flow and dotted arcs represent a compensation flow; d_1, d_2 and d_3 are the *delay*, *duration*, and *compensation duration* respectively. *Delay* is the time duration between two subsequent activities



Atomic Uncompensable Task Atomic Compensable Task

Fig. 2. Petri net representation of atomic tasks

(i.e., tasks). *Duration* is the maximum time required to finish a task. *Compensation duration* refers to the maximum time required to compensate a failed task. Delay, duration and compensation duration are expressed by integer values.

Both traditional control flows and control flows associated with compensable transactions can be easily edited and displayed graphically in the editor. The modeling elements are displayed in Fig. 3, using a notation similar to many workflow modeling languages. The editor produces workflow models which are *correct by construction* [31] which essentially means that incorrect composition of workflow activities is prevented. CWML$_T$ is a structured workflow modeling language which follows constraint-based approach and for this reason it becomes possible to not only guarantee that processes run correctly regarding their control and data flow, but also regarding the validity of the specified semantic constraints. As each workflow component has an underlying Petri net structure, the language has a sound mathematical foundation.

The editor is built as an Eclipse Plugin [6] using the Eclipse Graphical Editing Framework (GEF) [7]. Because of this architecture, the NOVA Editor is available in the development platform. Application developers can create models in a Java project, and generate workflow service classes from it (see section 3). Modeling, development and verification can be done in the same Eclipse Platform.

3 The NOVA Engine

The NOVA Workflow is developed using an SOA architecture. The engine was developed on the Spring and Hibernate platforms; Spring is a widely used open source framework that provides a consistent programming and configuration model that is well understood and used by developers worldwide. From the NOVA Editor, Workflow service classes are automatically generated to be deployed in the Spring container. These services are exposed to the outside world by service provider interfaces. The lifecycle of a workflow service bean (the class that contains the business logic) is managed by the spring container and at the time of instantiation, service beans register themselves to the workflow engine.

The NOVA Engine provides two modes for workflow engine integration: i) loosely coupled integration, ii) tightly coupled integration. The one to be selected depends on the system architecture. When the application services are not

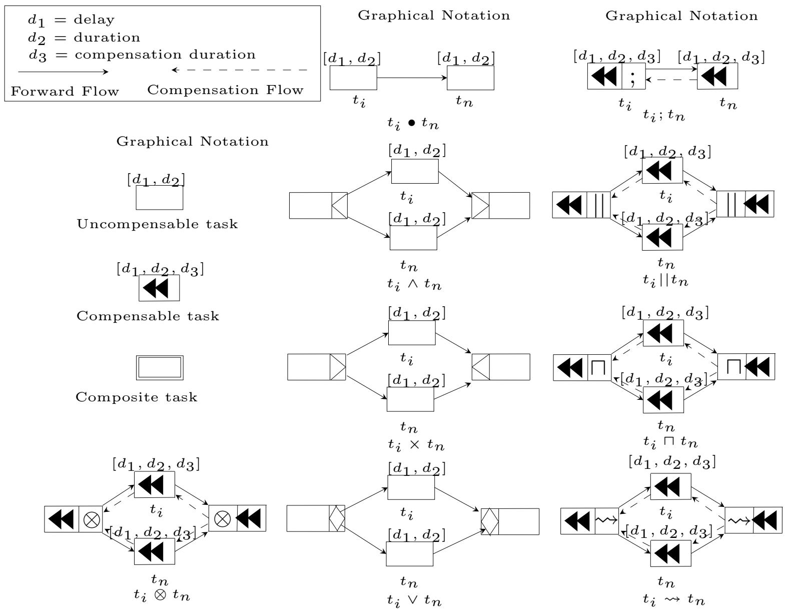


Fig. 3. Modeling Elements of CWML$_T$

deployed in the spring framework, loosely coupled integration should be selected; by this integration, workflow services are invoked by application services from outside the spring container. The NOVA Engine updates the task status when a particular service bean is invoked. On the other hand, if the application services are deployed in the spring container, tightly coupled integration is recommended; here application services are extended by workflow service classes.

Although the NOVA Workflow provides all support to develop a full swing client application, the current implementation of the NOVA Workflow does not generate default forms to take input from the user. In future, a form builder (or default form generator) will be incorporated with the NOVA Workflow. With the current version of NOVA Workflow, client applications may be developed in various platforms and they may communicate with the NOVA Engine using RMI, JSP, HTTPInvoker, WebService, etc. In future we intend to support greater mobility using sophisticated technologies such as iPads, Tablet PCs, etc. Accessibility and mobility are important for health care applications. It is anticipated that advanced mobile applications will improve outcomes as physicians, nurses and other clinicians can access both recent and historical information while visiting a patient. Advanced interoperability vis a vis international standards such as HL7, OpenEHR is on the horizon for future research.

4 The NOVA Translator

Once a workflow is designed with compensable tasks, its properties can be verified by model checkers such as SPIN, SMV or DiVinE. Modeling a workflow with the input language of a model checker is tedious and error-prone. In [30] we provided a manual translation from CWML to DVE the input language of the DiVinE model checker, which was extended to an automated translator to CWML$_T$ in [27]. This automated translation method has been integrated into the NOVA Workflow. Now, using the Editor one can graphically design a workflow using the CWML$_T$ and write the business logic for the tasks (in Java); then the translator automatically translates the model to DVE (see [30,27]).

However, the time required for the verification was often unacceptable. Our experiments showed that even though DiVinE is equipped with the Partial Order Reduction technique and several different model checking algorithms, it required a great deal of memory and time for the verification of large models. In [29] we developed a model reduction algorithm for the models built using (untimed) CWML. The algorithm has been implemented in the NOVA Translator, which takes a workflow model and the specification of an LTL$_{-X}$ property ϕ and reduces the model, based on the property ϕ. The proof of the stuttering equivalence of the original and reduced models may be found in [29]; thus the truth of ϕ is preserved and reflected. A demonstration of the effectiveness of the proposed method in reducing the size of the state space may be found in [29]. We expect the proposed algorithm can be easily applied to any block-structured modeling language (e.g., ADEPT2 [31]), and currently we are extending the reduction to models built using CWML$_T$. The reduction algorithm incorporates the feature of data-awareness currently found in many workflow modeling languages. Other reduction algorithms involving time and including the notion of *leaping time* [37] are currently under investigation. An extention of the translator to other model checkers, i.e., SMV, SPIN, Maude are under development which will allow us to provide an in-depth performance comparison.

5 The NOVA Monitor

Workflow monitoring is an active research area which has great importance especially in safety critical systems for enforcing policies, and achieving effciency and reliability goals. Monitoring is a frequent requirement in healthcare environments where monitor systems are typically configured to notify clinicians about abnormalities in a process. Designing a monitor for a time constrained compensable workflow in a healthcare setting is complex and it must be verified to ensure it operates properly before its use with a patient.

The NOVA Monitor uses a graphical modeling language for monitoring workflows which is based on Time Petri nets [14], and integrates such time constrained monitors (called WMon-nets) with workflow models (called CWF-nets) built using CWML$_T$. Fig. 4 shows the graphical notation for workflow monitor components. A green transition is associated with the forward transition (pt_1 in Fig. 2) of an atomic (uncompensable or compensable) task in a CWF-net, and a yellow transition is associated with the compensation transitions (pt_2, pt_3 in Fig. 2) of an atomic compensable task in a CWF-net.

Green and yellow transitions are virtual transitions and do not execute like ordinary transitions in a Petri net. It is through these virtual transitions that we integrate a WMon-net with a CWF-net. The output places of green transitions and yellow transitions get tokens when the corresponding transitions (pt_1, pt_2, pt_3) execute in a CWF-net (see Fig. 5).

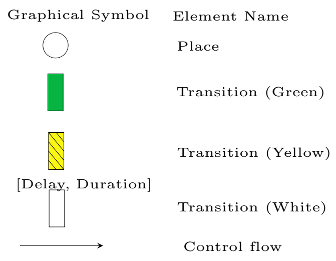


Fig. 4. Graphical representation of monitor components

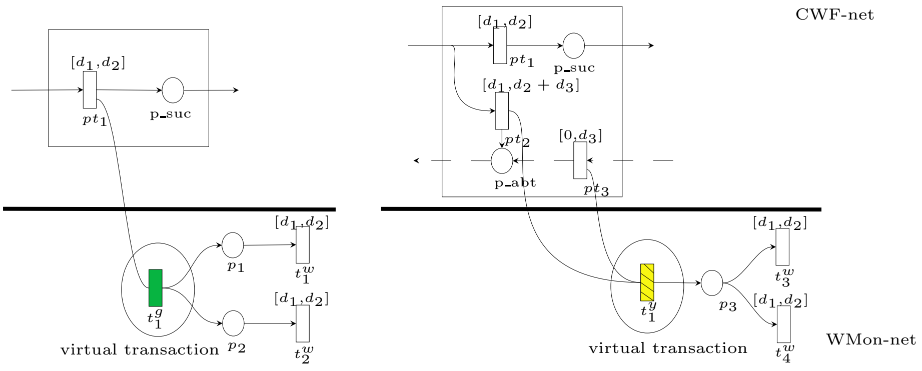


Fig. 5. A virtual transition

In order to verify properties of a monitor (modeled as a WMon-net) we combine the state space of the monitor with the state space of the workflow. A monitor monitors the processes of a workflow so it cannnot generate its own state space without input from that workflow. From the generated state space, the model checker can verify the properties of the system. In the case study, we show a compensable workflow with a monitor and verify properties of the workflow including properties to show that the monitor is working correctly. An ontology was incorporated with the monitor; results of complex queries to the ontology can guide the control flow. It is anticipated that the workflow monitor may be integrated with other workflow languages, with Petri net based foundations. Details of the monitor system may be found in [4].

6 The NOVA Browser

Usability is an important quality in any product but is often neglected in software products. It was found that EHR (Electronic Health Record) usability is at the root of many medical errors [21]. For healthcare systems, usability is not an expectation; rather it is an essential requirement, as patient safety is involved and clinician's time is a valuable resource. We now present a user friendly browser, called the NOVA Browser, for EHR with which health professionals can view a patient's medical information without losing the patient's context while browsing. The flexibility and capacity for brainstorming enhances the user experience. The browser also incorporates a time travel view and a chart view that helps health professionals observe and monitor a patient's medical condition. The unique time travel view of the browser makes it a helpful tool for cause and effect analysis.

6.1 Current Features of the NOVA Browser

Hierarchical Representation of Data. The NOVA Browser hierarchically represents a patient's EHR in a mind map. A mind map [17] is a graphical way to represent ideas and concepts. The nodes (i.e., ideas, concepts, items, etc) are represented hierarchically in a mind map. A mind map (as opposed to traditional notes or text) structures information in a way that resembles much more closely how the brain actually works. Since it is an activity that is both analytical and artistic, it engages the human brain in a rich way, helping in all its cognitive functions. While pictorial methods for recording knowledge and modeling systems have been used for centuries in learning, brainstorming, memory, visual thinking, and problem solving by educators, engineers, psychologists, and others, its use in software systems to provide a means to involve the user more with the system is rare (e.g., mind map has been used in OpenEHR as an archetype [1]).

Fig. 6 shows an EHR representation in the NOVA Browser. The browser displays the records of a particular patient's case. The centre of the map shows the case number (Case-1). The clinician can unfold any of the branches and can view the details of the case. The browser performs a database query using the case number and loads the records from database tables or views. The NOVA Browser provides a case selection window with which the clinician can switch to a different patient's case.

Time Travel View. In order to enable the time travel view, the database tables include the Timed Table, to preserve the historical information and time. In a Timed Table, no records are deleted or updated by replacing the original record; instead of an update operation to a row, a new entry with status UPDATE is inserted into the table and a column is used to indicate the parent record, whose information is being updated. When displaying the records in the browser, only the latest records are shown. When the user travels back, the browser fetches historical records and displays them in the map.

The time travel view provides an easy way for clinicians to go back to when a certain record was inserted or updated and then check its effect by travelling forward from that time. The browser provides four types of time travel: Travel Backward (or Travel Forward) to a past (or future) time when a selected record was inserted, updated or deleted (in this case the user needs to select a node), and Travel One Step Backward (or Forward), which is travel to the previous

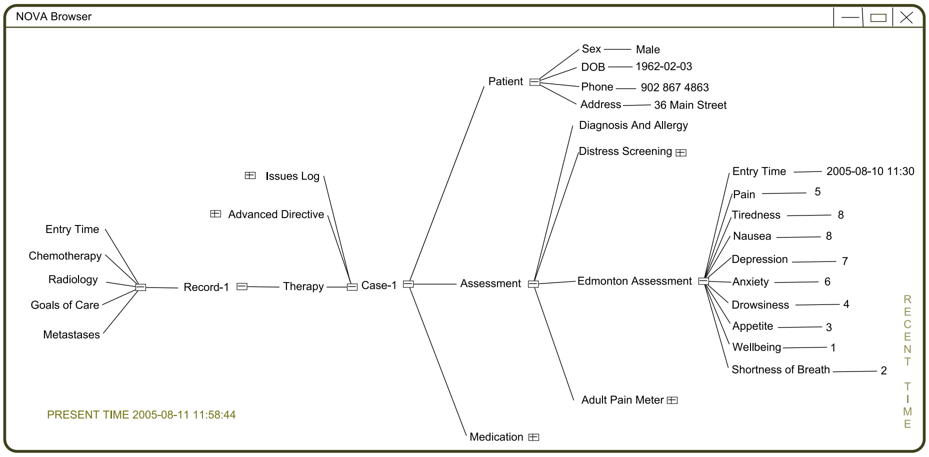


Fig. 6. Hierarchical Data representation in NOVA Browser

(or next) time that a record was inserted or updated for the selected case (in this case the user does not select a node and the search operation is performed globally on all tables for the case).

If the clinician selects the ‘Assessment’ node and travels back, the browser will jump to the time when an Assessment record was inserted or updated. Note that Assessment is an abstract base class with three concrete subclasses (i.e., Distress Screening, Edmonton Assessment, Adult Pain Meter). The NOVA browser uses hibernate [9] to communicate with the database. As hibernate is an Object Relational Mapper, any query performed on a base class is executed on all its subclasses. The NOVA Browser displays a transition from one map to another by doing animation about the Z-axis, which gives the impression of travelling backward or forward.

Chart View. Charts and graphs play an important role for analyzing information. Detecting patterns in patient populations is an esential function in the administration of health care systems. The visual representation of complex information can help process large amounts of data to detect and observe such patterns. However it is very difficult to pre-configure all the charts with all the different combinations of parameters which may be needed by clinicians and administrators. The NOVA Browser incorporates a chart view allowing the user to select the desired chart parameters. The clinician selects some nodes from the browser, adds them to the parameter list and then selects a time range. The chart viewer generates the chart using those selected parameters by plotting time on the X-axis and the parameters on the Y-axis. As a clinician can select any node from the browser, the browser will either plot the exact values of the parameter or present them symbolically.

6.2 Proposed Application of the NOVA Browser

We propose applying the NOVA Browser to process diagnosis. To provide flexibility to the system, we allow two ways of interacting: i) Workflow based, ii) Task based. The workflow system will provide a worklist window from which the user can select a task and execute it. Alternatively, the user may skip the workflow and can fill in and submit a form related to a task. A workflow provides better support but is less flexible. We will allow the user to skip the workflow, as initially all patient case scenarios are not known so the workflow should not restrict the user to perform an emergency job. The workflow may be adjusted later on through a redesign.

All the activities performed by the user are recorded in an event log which will be used to restructure the workflow. Two types of events may be found in the event log: i) a workflow event for a task being executed from the worklist, and ii) an ad-hoc event for a form being executed in an ad-hoc manner. We have addressed the issue of an evolutionary process restructuring through a process diagnosis mechanism. We propose a new design of process diagnosis in the NOVA Workflow which incorporates a process mining technique using the NOVA Browser.

Motivation of the Work. The term process mining refers to methods for distilling a structured process description from a set of real executions. In [20], Cook and Wolf described three methods for process discovery: one using neural networks, one using a purely algorithmic approach, and one using a Markovian approach. The authors considered the latter two as the most promising. The purely algorithmic approach builds a finite state machine (FSM) where states are fused if their futures (in terms of possible behavior in the next k steps) are identical. The Markovian approach uses a mixture of algorithmic and statistical methods and is able to deal with noise. However, they did not provide an approach to generate explicit process models. The idea of applying process mining in the context of workflow management was first introduced in [12]. This work is based on workflow graphs, which are inspired by workflow products such as IBM MQSeries workflow (formerly known as Flowmark) and InConcert.

Van der Aalst et. al., studied a number of process mining or workflow mining techniques in [36] and pointed out two problems. The first is to find a workflow graph generating events appearing in a given event log and the second is to find the definitions of edge conditions (i.e., pre-conditions). In [36], they provided a concrete algorithm for tackling the first problem.

For a simple system with a few tasks and enough workflow logs it is quite easy to construct a process model, but for more realistic situations (e.g., healthcare) there are a number of complicating factors [36]:

1. Mining is difficult in large workflow models and has a high degree of complexity if the model exhibits alternative and parallel routing (in which case the workflow log will typically not contain all possible combinations).

2. Workflow logs will typically contain noise, i.e., parts of the log may be incorrect, incomplete, or refer to exceptions. For example, events can be logged incorrectly because of human or technical errors.
3. Events can also refer to rare or undesirable events. Consider, for example, a workflow in a hospital. If, due to time pressure, the order of two events (e.g., make X-ray and remove drain) is reversed, this does not imply that this would be part of the regular medical protocol and should be supported by the hospital's workflow system. Also two causally unrelated events (e.g., take blood sample and death of patient) may happen in a sequence without implying a causal relation. Exceptions which are recorded only once should not automatically become part of the regular workflow.

Process Diagnosis Using the NOVA Browser. For a safety critical system it is not desirable to deploy the workflow model discovered by the automated process mining tool without a proper validation by domain experts for the workflow model discovered by process mining. The conditions of *XOR*, *OR*, *Loop* discovered by a process mining tool from the event log may not be the exact condition for the selection of the branches. For a healthcare application there are numerous parameters that guide the flow of the tasks.

The proposed workflow management life cycle consists of 6 steps (i.e., 1. Workflow Design, 2. Workflow Validation, 3. Workflow Enactment 4. Event Log 5. Process Mining 6. Process Restructure), where step 5, 6 and 2 are part of the process diagnosis. Consider the workflow fragment 'Medicine Administration' as shown in Fig. 7. The initial design of the Loop condition was to *iterate until the dose is finished*. During the execution of the workflow it was found that a patient was allergic to certain drug and the 'Medicine Administration' task was cancelled for this reason.

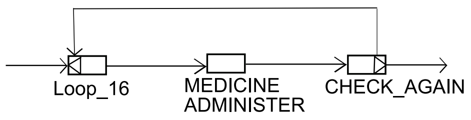
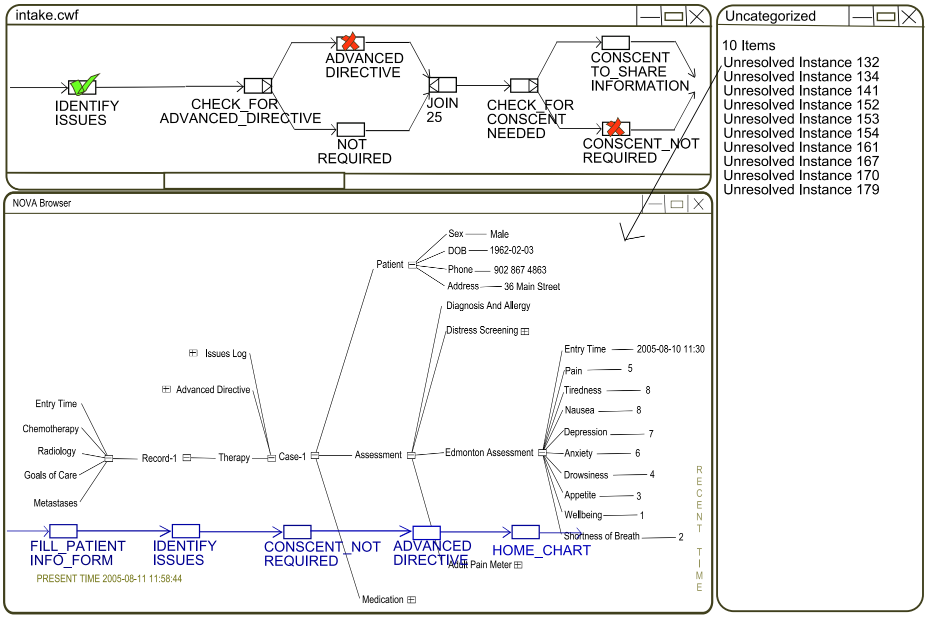


Fig. 7. Medicine Administration

Let us assume that the allergy information was inserted into the system and another change in the patient's medical condition happened at the same time; for example, the patient's PPS (Pulse Per Second) value reduced. Here the decrement of the patient's PPS value is noise which makes it hard to identify the exact Loop condition for a process mining tool.

We propose a design for a 'Process diagnosis tool' using the NOVA Browser to handle these problems. The proposed system is shown using the NOVA Editor in Fig. 8, where the workflow model discovered by the process mining tool can be restructured by the user easily and efficiently. The workflow component's view is shown at the bottom left side of Fig. 8. The user makes some changes to the workflow model, and invokes an analyze action to the system; by analyzing the



10 Items
Unresolved Instance 132
Unresolved Instance 134
Unresolved Instance 141
Unresolved Instance 152
Unresolved Instance 153
Unresolved Instance 154
Unresolved Instance 161
Unresolved Instance 167
Unresolved Instance 170
Unresolved Instance 179

Sex	Male
DOB	1962-02-03
Phone	902 867 4863
Address	36 Main Street

Entry Time
Chemotherapy
Radiology
Goals of Care
Metastases

Entry Time	2005-08-10 11:30
Pain	5
Tiredness	8
Nausea	8
Depression	7
Anxiety	6
Drowsiness	4
Appetite	3
Wellbeing	1
Shortness of Breath	2

PRESENT TIME 2005-08-11 11:58:44

Fig. 8. Process diagnosis using the NOVA Browser

event logs, the system searches for the workflow instances which do not satisfy the redesigned workflow model. The unresolved workflow instances are shown at the top right side of the editor. The user opens an unresolved workflow instance in the NOVA Browser (shown in the middle of the editor) to analyze the case. The sequence of the tasks' execution for the selected case is shown at the bottom of the NOVA Browser. The user can use the time travel view in the Browser to go back to past records and see under what condition the tasks were executed. For example, consider the above 'Medicine Administration' problem; the user can go back in time to when the drug administration was stopped, find the allergy recorded at that time and then edit the workflow and restructure it with the exact condition. With such a mining and reconfiguring feature, a WfMS may be considered to be adaptive.

7 Case Study

We model a workflow and a monitor system following the guidelines for the management of cancer related pain in adults [16]. If a patient is taking a strong opioid, a pain reassessment should be done after a certain interval. The guideline for the strong opioid regimen says that if a patient is responding (i.e., current pain level is less than previous pain level) then another reassessment should be done within a week; if a patient is not responding then it suggests a different reassessment interval depending on the current pain level. The guideline suggests 'Opioid toxicity' or the 'Continuation of dose titration' depending on the 'Response'. 'Management of side effects' is a compensation for these processes.

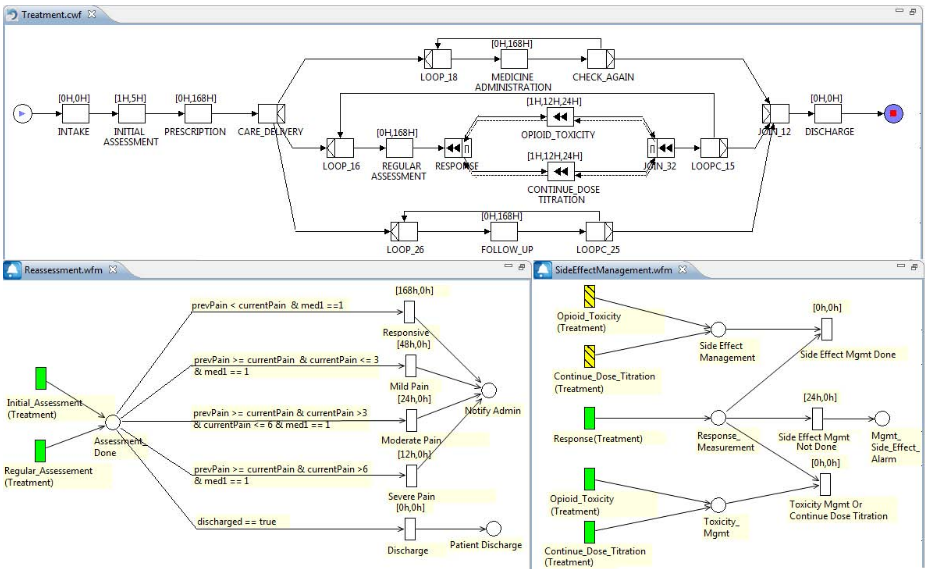


Fig. 9. “Treatment workflow”, “Reassessment monitor”, and “Side effect management monitor” for patients

Fig. 9 shows the treatment workflow (a time constrained CWF-net) at the top, the reassessment monitor (a WMon-net) at the bottom left which integrates with both the INITIAL_ASSESSMENT and REGULAR_ASSESSMENT tasks, and the side effect management monitor (a WMon-net) at the bottom right which integrates with the compensable tasks OPIOID_TOXICITY and CONTINUE_DOSE_TITRATION. In this model, when a patient is admitted (task INTAKE is executed) to the hospital an initial assessment is done (task INITIAL_ASSESSMENT is executed) where the patient's pain level is recorded. A physician prescribes medicine for the patient (related task is PRESCRIPTION) which may be updated at Follow up. While the patient is taking his medicine (task MEDICINE_ADMINISTRATION executes), a regular assessment is done (task REGULAR_ASSESSMENT is executed) after certain interval concurrently with other processes, e.g., Medicine Administration, Follow up. Note that, CARE_DELIVERY is an *AND-Split* task which activates all of its outgoing branches and RESPONSE is an *Internal choice split* task which activates one of its outgoing branches during execution. OPIOID_TOXICITY and CONTINUE_DOSE_TITRATION are compensable tasks and they compensate for any side effects found during execution.

The monitor system observes the interval of assessment and notifies the clinician if another reassessment is not done within the time suggested by the guidelines. The general knowledge base for medicine is very large, and frequently organized as a medication ontology. To show how our system can integrate with an ontology, we designed a small ontology in OWL 2.0 representing the facts and rules about strong opioids used for querying reassessment time. We integrated the ontology with the monitor system using the FaCT++ reasoner. The system generates a query with the list of medication that a patient is taking, and the current and previous pain levels as parameters and sends them to the FaCT++ reasoner. The reasoner computes whether the medicine is a strong opioid and returns the suggested reassessment time. Some of the properties we verified are provided below with their LTL formulas:

– If a patient is under the strong opioid regimen, not responding (to medication), current pain level is > 6 and another reassessment is not done within 12 hours, the clinician will get a notification (In LTL, G ((strong_opioid_patient && patient_is_not_responsive && cur_p_level_gr_than_six && reassessed_twelve_hrs_ago) – > F (clinician_is_notified))).
– If a patient is discharged, the clinician will not receive any reassessment alert (In LTL, G (patient_discharged – > ! F (clinician_is_notified))).
– If required 'Side effects' are not managed within 24 hours, the clinician will get a notification (In LTL, G (response_measured && side_effect_not_managed && resp_measured_24_hrs_ago – > mgmt_side_effect_alarm))

The results of the model checking showed that first two properties were false, and provided counter examples. For the first property the counter example says if those conditions are true then the clinician may not get a notification if the patient is discharged. For the second property, the counter example says the clinician receives the notification if the discharge operation executes at the same time as a notification was supposed to be sent. It is clear from the counter examples that there exists a flaw in the models; we determined that the patient's discharge was not taken into consideration in the pre-conditions of *Responsive*, *Mild Pain*, *Moderate Pain* and *Severe Pain* transitions. As a result while the *Discharge* transition is ready, other transitions could possibly be ready and execute. The initial model was corrected by rewriting the pre-conditions and subsequent model checking showed that both properties were satisfied. Due to space limitation we presented a small case study here which involves the modeling of both a timed Compensable Workflow, with monitors and the translation to DVE and model checking. The workflow was executed in a J2EE server and interfacing to the Ontology was done by the FACT++ reasoner. Altogether, we used the NOVA Editor, Engine, Translator and Monitor. Interested readers are referred to [29] where they will find details of a much larger case study involving the reduction.

8 Related Work

Petri nets [28] is a popular formalism for the design of concurrent systems because of its sound mathematical foundations. Many analysis techniques are available for Petri nets. Workflows may be designed by Petri net tools such as TINA [15], Romeo [33], etc. Reo [24] is a graphical channel-based coordination language that enables the modeling of complex behavioral protocols using a small set of channel types with well-defined behavior. Designing a large workflow model with these tools ([15,33,24]) is difficult to manage. Designing a workflow with compensation using these tools is particularly complex as the model becomes very large; the use of a high level modeling language is preferable. These tools, moreover do not use reduction techniques explicitly to verify a large model. In [22] the authors provided a reduction technique for Coloured Petri nets. These techniques preserve the liveness of the model and any LTL formula that does not observe the reduced transitions of the net. This approach to reduction differs from ours [29]; in our approach we take the model (M) and the property (ϕ) both into consideration and reduce the model in such a way that the reduced model, $M' \vDash \phi$ *iff* the original model $M \vDash \phi$. Another difference between [22] and our approach is merging vs. reducing transitions. We follow the later approach which reduces tasks, pre-conditions, actions and variables from the original model. It will significantly reduce the memory size for each state; as a result the memory size of the whole state space becomes less.

UPPAAL [13] is a popular timed automata model checker but the distributed version of UPPAAL is under development. The data aware verification method we presented here using the parallel distributed model checker DiVinE provides excellent support to verify large systems.

Some popular workflow management systems are YAWL [35], ADEPT2 [31], BPEL [2], etc. For workflow modeling YAWL uses workflow patterns but is an unstructured language; on the other hand, ADEPT2 uses a block-structured language. The use of structured vs. unstructured workflow language is debatable; usually unstructured workflow languages are more expressive than block-structured languages but the soundness is not guaranteed by construction as in a block-structured language. CWML$_T$ is a block-structured language and hence the soundness is guaranteed by construction (proof in [30]). YAWL comes with limited forms of verification (e.g., livelock, deadlock, etc). In [38] the authors provided reduction rules for YAWL workflows with *Cancellation* regions and *OR-joins* to reduce the size of the workflow, while preserving its essential properties with respect to a particular analysis problem. Here the authors only focused on the soundness analysis, whereas our reduction method works for any LTL$_{-X}$ property (The subset of LTL formulas not containing the X operator).

An ADEPT2 workflow can be verified using the SeaFlows compliance checker [23]. In [23] the authors discussed an abstraction approach which can serve as a preprocessing step; this is an efficient way to deal with the state explosion problem. This strategy is orthogonal to our strategy; our methods can be further improved by the automated abstraction technique to reduce the data domain for variables and their method can be improved by the reduction technique we discuss here.

YAWL and ADEPT2 do not have compensation mechanisms, whereas BPEL has a built-in compensation for atomic processes (there are no compensable operators). BPEL has been used in the industry for some time and there are many publicly available tools (e.g., BPEL2PN [3], WSEngineer [11]) to analyze workflows designed in BPEL. BPEL2PN does not provide data aware verification and WSEngineer does not use any explicit reduction technique. Although BPEL has a built-in compensation, it is not t-calculus based. CWML$_T$ is more expressive and gives us flexibility in designing compensable workflow with its rich semantics.

Simmonds et al. presented the tool RuMoR in [32] which performs monitoring of web service applications, and, when violations are discovered, automatically proposes and ranks recovery plans. Properties, specified using property patterns, are transformed into finite state automata. While runtime monitoring, some compensation mechanism, verification are common to our system there are differences. RuMoR takes a BPEL program as input and translates to a labeled transition model using WS-Engineer. Monitors are specified as finite-state automata. Although data aware verification can be done in RuMoR it has limitations with respect to time. RuMoR was implemented within the IBM WebSphere using the interception mechanism, whereas the architecture of NOVA Workflow is light-weight as it uses Spring and aspect oriented programming techniques which enabling its use with various J2EE application servers. In addition, NOVA Workflow uses an ontology for decision support.

PROM [8] is an automated process mining tool which requires enough event logs to extract process flows; here in this paper we proposed another process mining tool which is semi-automated; the proposed tool is designed to be user friendly as it incorporates the NOVA Browser.

Declare [34] is a prototype of a workflow management system which follows a declarative approach to business process modeling and execution. Unlike conventional systems, which use graph-like modeling languages (e.g., Petri-nets), Declare uses logic (i.e., LTL) to model and execute business processes. Modeling time, compensation and monitors in Declare is complex and its formal verification needs further research. Moreover, many features of NOVA Workflow (e.g., runtime monitoring, verification, browser, process diagnosis, etc) are not covered in Declare.

NOVA, meaning 'new' in Latin, summarizes the four important features of the framework: compeNsation, Ontology, Verification and Adaptability. We have developed an interface to permit the workflow engine to consult an Ontology knowledge base to guide the execution of the workflow engine. This paper also presents a new technology for analyzing and visualizing healthcare information that will significantly improve the usability of EHR systems. In future we will interface the system with various laboratory and radiology equipment to display a patient's reports in the browser and develop an automated form builder. We will further support workflow flexibility for instance, by incorporating a sophisticated client application.

Acknowledgments. We acknowledge support from the Atlantic Canada Opportunities Agencies, the Natural Sciences and Engineering Research Council of Canada, the Canadian Foundation for Innovation, and the Nova Scotia Research and Innovation Trust. We are grateful for the enthusiastic support of clinicians and administrators from Guysborough Antigonish Strait Health Authority (GASHA).

References

1. OPENEHR, http://www.openehr.org/home.html (last accessed, July 2011)
2. IBM, BEA, Microsoft, SAP, Siebel. Business process execution language for web services version 1.1 (May 2003)
3. BPEL2PN, http://www2.informatik.hu-berlin.de/top/bpel2pn/ (last accessed, April 2011)
4. Rabbi, F., Mashiyat, A., MacCaull, W.: Model checking workflow monitors and its application to a pain management process. In: Proceedings of 1st International Symposium on Foundations of Health Information Engineering and Systems, FHIES 2011, Johannesburg, South Africa, pp. 110–127 (2011)
5. DiVinE project, http://divine.fi.muni.cz/ (last accessed, October 2011)
6. Eclipse plugin, http://www.eclipse.org/articles/article-plug-in-architecture/plugin_architecture.html/ (last accessed, April 2011)
7. Graphical Editing Framework, http://www.eclipse.org/gef/ (last accessed, April 2011)
8. PROM, http://www.processmining.org/ (last accessed, May 2011)
9. Relational persistence for java and .net, http://hibernate.net/ (last accessed, May 2011)
10. Spring framework, http://www.springsource.org/ (last accessed, May 2011)
11. WSENGINEER, http://www.doc.ic.ac.uk/ltsa/eclipse/wsengineer/ (last accessed, July 2011)
12. Agrawal, R., Gunopulos, D., Leymann, F.: Mining Process Models from Workflow Logs. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) EDBT 1998. LNCS, vol. 1377, pp. 469–483. Springer, Heidelberg (1998)
13. Bengtsson, J., Larsen, K.G., Larsson, F., Pettersson, P., Yi, W.: UPPAAL — a Tool Suite for Automatic Verification of Real–Time Systems. In: Alur, R., Sontag, E.D., Henzinger, T.A. (eds.) HS 1995. LNCS, vol. 1066, pp. 232–243. Springer, Heidelberg (1996)
14. Berthomieu, B., Diaz, M.: Modeling and verification of time dependent systems using time Petri nets. IEEE Trans. Softw. Eng. 17, 259–273 (1991)
15. Berthomieu, B., Vernadat, F.: Time petri nets analysis with Tina. In: QEST, pp. 123–124 (2006)
16. Broadfield, L., Banerjee, S., Jewers, H., Pollett, A., Simpson, J.: Guidelines for the management of cancer-related pain in adults. Supportive Care Cancer Site Team, Cancer Care Nova Scotia (2005)
17. Buzan, T.: The Mind Map Book. Penguin Books (1996)
18. Clarke, L.A., Chen, Y., Avrunin, G.S., Chen, B., Cobleigh, R.L., Frederick, K., Henneman, E.A., Osterweil, L.J.: Process programming to support medical safety: A case study on blood transfusion. In: ISPW 2005, pp. 347–359 (2005)
19. Combi, C., Posenato, R.: Controllability in Temporal Conceptual Workflow Schemata. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 64–79. Springer, Heidelberg (2009)
20. Cook, J.E., Wolf, A.L.: Discovering models of software processes from event-based data. ACM Trans. Softw. Eng. Methodol. 7, 215–249 (1998)
21. Edwards, P.J., Moloney, K.P., Jacko, J.A., Sainfort, F.: Evaluating usability of a commercial electronic health record: A case study. Int. J. Hum.-Comput. Stud. 66, 718–728 (2008)

22. Evangelista, S., Haddad, S., Pradat-Peyre, J.-F.: Syntactical Colored Petri Nets Reductions. In: Peled, D.A., Tsay, Y.-K. (eds.) ATVA 2005. LNCS, vol. 3707, pp. 202–216. Springer, Heidelberg (2005)
23. Knuplesch, D., Ly, L.T., Rinderle-Ma, S., Pfeifer, H., Dadam, P.: On Enabling Data-Aware Compliance Checking of Business Process Models. In: Parsons, J., Saeki, M., Shoval, P., Woo, C., Wand, Y. (eds.) ER 2010. LNCS, vol. 6412, pp. 332–346. Springer, Heidelberg (2010)
24. Kokash, N., Krause, C., de Vink, E.P.: Time and data-aware analysis of graphical service models in Reo. In: Proceedings of the 2010 8th IEEE International Conference on Software Engineering and Formal Methods, SEFM 2010, pp. 125–134. IEEE Computer Society, Washington, DC (2010)
25. Li, J., Zhu, H., He, J.: Specifying and Verifying Web Transactions. In: Suzuki, K., Higashino, T., Yasumoto, K., El-Fakih, K. (eds.) FORTE 2008. LNCS, vol. 5048, pp. 149–168. Springer, Heidelberg (2008)
26. Li, W., Fan, Y.: A time management method in workflow management system. In: Grid and Pervasive Computing Conference, pp. 3–10 (2009)
27. Mashiyat, A.S., Rabbi, F., MacCaull, W.: Modeling and Verifying Timed Compensable Workflows and an Application to Health Care. In: Salaün, G., Schätz, B. (eds.) FMICS 2011. LNCS, vol. 6959, pp. 244–259. Springer, Heidelberg (2011)
28. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE 77(4), 541–580 (1989)
29. Rabbi, F.: Design, development and verification of a compensable workflow modeling language. MSc Thesis, St. Francis Xavier University (2011)
30. Rabbi, F., Wang, H., MacCaull, W.: Compensable WorkFlow Nets. In: Dong, J.S., Zhu, H. (eds.) ICFEM 2010. LNCS, vol. 6447, pp. 122–137. Springer, Heidelberg (2010)
31. Reichert, M., Rinderle, S., Kreher, U., Acker, H., Lauer, M., Dadam, P.: ADEPT2 - next generation process management technology. In: Proceedings Fourth Heidelberg Innovation Forum, Aachen, D.punkt Verlag (April 2007)
32. Simmonds, J., Ben-David, S., Chechik, M.: Guided recovery for web service applications. In: SIGSOFT FSE, pp. 247–256 (2010)
33. Traonouez, L.-M., Lime, D., Roux, O.: Parametric model-checking of stopwatch petri nets 15(17), 3273–3304 (2009)
34. van der Aalst, W., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. Computer Science - Research and Development 23, 99–113 (2009)
35. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: yet another workflow language. Information Systems 30(4), 245–275 (2005)
36. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow mining: a survey of issues and approaches. Data Knowl. Eng. 47, 237–267 (2003)
37. Wang, H., MacCaull, W.: An efficient explicit-time description method for timed model checking. In: PDMC, pp. 77–91 (2009)
38. Wynn, M.T., Verbeek, H.M.W.E., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Reduction rules for YAWL workflows with cancellation regions and OR-joins. Information and Software Technology 51(6), 1010–1020 (2009)

Experiences with a Compositional Model Checker in the Healthcare Domain

Jozef Hooman[1,2], Robert Huis in 't Veld[3], and Mathijs Schuts[3]

[1] Embedded Systems Institute, Eindhoven, The Netherlands
[2] Radboud University, Nijmegen, The Netherlands
[3] Philips Healthcare, Best, The Netherlands

Abstract. This paper describes the use of a formal method to support component-based development in the healthcare domain. The method is based on a commercial tool suite which combines formal modeling, compositional model checking, and code generation. The main approach of the tool suite will be explained and demonstrated from a user point of view. We report about experiences with this approach at the company Philips Healthcare for the design of control software for advanced interventional X-ray systems. This concerns formal interface definitions between the main system components and detailed design of control components.

Keywords: formal modeling, component-based development, model checking, code generation, industrial experience.

1 Introduction

We report about experiences at Philips Healthcare with a commercial tool based on formal methods. The tool is used for the development of control software of interventional X-ray systems. These systems are used for minimally invasive surgery, e.g., improving the throughput of a blood vessel by placing a stent via a catheter where the surgeon is guided by X-ray images. These techniques avoid open heart surgery. This has many benefits in the healthcare domain such as improved productivity, more effective treatments, better success rate, and increased quality of the life of patients. Moreover, this type of image-guide non-invasive surgery leads to lower healthcare costs by shorter hospital stays and higher throughput.

An example of such a system is depicted in Fig. 1, showing a patient table, a C-arm with X-ray tube and detector, and a large screen for images and user guidance. This equipment is placed in the so-called intervention room, where physicians press pedals and operate switches to control the acquisition of X-ray images. In addition, there is an additional control room where medical personnel can view and store images, write reports, and prepare the intervention. Preparation includes, for instance, entering patient data and details about the treatment that determine the amount of X-ray and the movements of the C-arm.

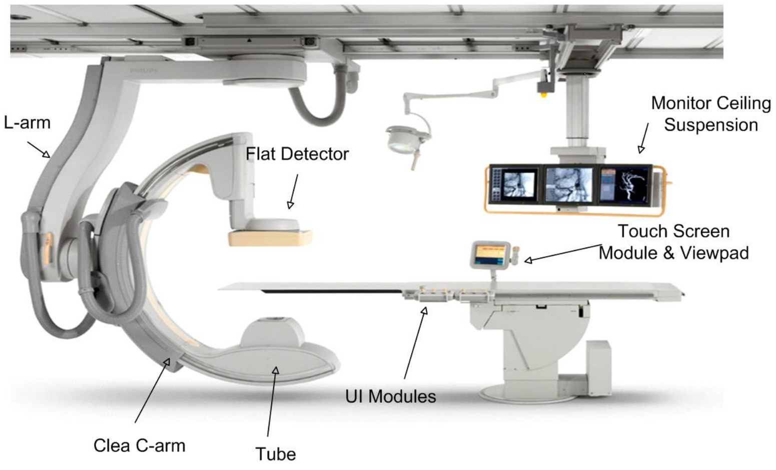


Fig. 1. Interventional X-ray system

To support new medical procedures, it is important to be able to incorporate medical innovations fast in such systems. This is challenging, because high quality standards have to be met. This requires a software architecture that can be adapted and extended easily without long test and integration times, while still maintaining the current high-level of quality.

To meet these goals, Philips Healthcare is migrating towards a component-based software architecture with clearly defined interfaces. Interfaces are defined formally using a formal tool called ASD:Suite of the company Verum [18]. This tool is also used to implement control components of the new architecture based on formally verified design models. The use of this formal technique at Philips Healthcare is motivated by the aim to remove faults as early as possible during the software development process, thus reducing the test and integration time, which is often long and unpredictable. We describe our experiences when applying this approach to a part of an interventional X-ray system.

ASD:Suite of Verum is a development tool that embeds the Analytical Software Design (ASD) technology into a software design environment. ASD [3,13] enables the application of formal methods into industrial practice by a combination of the Box Structure Development Method [15] and CSP [11]. To obtain complete and consistent specifications, a Sequence-Based Specification Method [16] is used where the response to all possible sequences of input stimuli has to be defined. Sequences that cannot happen must be declared illegal explicitly. The sequence-based specifications have been translated into CSP and the FDR model checker [8] is used to verify refinement steps. ASD:Suite hides the CSP details and provides the user with a tabular notation to describe state machines, a standard set of correctness checks, and a nice visualization of error traces.

The ASD approach distinguishes two types of models which are both described by a similar tabular notation: *interface models* and *design models*. The tool ASD:Suite allows code generation from design models for a number of programming languages (C, C++, C#, Java). A design model implements a certain interface model, typically using services of other components by referring to their interface models. The model checker of ASD:Suite verifies that calls to these used components are correct with respect to their interface models. Moreover, it is checked that the design model conforms to the implemented interface model. The approach is compositional [12], since the verification uses only the interfaces of the used components, without knowing their implementation. It is also not needed that these used components are realized using ASD. Since the ASD approach is intended for control components, used components that involve data manipulations or algorithms will be implemented by other techniques; such components are called *foreign components*.

Related to ASD:Suite are formal methods with commercial tool support and code generation from formal models. For instance, the industrial tool VDM-Tools [5] contains a code generator for the formal language VDM++ [7]. The B-method [1], which has been used to develop a number of safety-critical systems, is supported by the commercial Atelier B tool [4]. The SCADE Suite [6] provides a formal industry-proven method for critical applications with both code generation and verification. Compared to ASD, these methods are less restricted and, consequently, correctness usually requires interactive theorem proving. ASD is based on a careful restriction to data-independent control components to enable fully automated verification.

This paper is structured as follows. In Sect. 2, the interface models of ASD are explained using a small camera example which resembles a medical imaging device. The use of interface models at Philips Healthcare is described in Sect. 3. Sect. 4 introduces the design models of ASD and model checking, again using the camera example. The application of these models at Philips is discussed in Sect. 5. Concluding remarks can be found in Sect. 6.

2 ASD Interface Models

To illustrate the ASD approach we use a small camera example. We start with an ASD interface model which represents the traces a server offers to its clients. There are two ways of communication between client and server:

- *Procedure calls* from client to server, which are synchronous in the sense that the client has to wait until the server is ready to accept the call. Next the client is blocked until the server returns the call. There are two types of calls:
  - Void calls, which return a void reply to signal the completion of the call
  - Valued calls, which return a value upon completion
- *Callbacks* from server to client, which are asynchronous events that can be sent by the server immediately.

To model the interface, e.g., to trigger callbacks, an interface model may also contain:

- *Internal modeling events*, which can be optional (meaning that they may happen) or inevitable (meaning that they will happen eventually).

For the camera example we have the following sets of calls, callbacks, and internal events:

- APICamera contains three calls:
  - PowerOn(): valued, with two possible return values: OnOK, OnFailed
  - PowerOff(): void
  - Click([in]exposureTime:int): void
- CBCamera contains four callbacks:
  - CBPicture([in]photo:image)
  - CBOn()
  - CBEmptyBattery()
  - CBOnFailed()
- INTCamera contains four internal modeling events to trigger the four callbacks above:
  - PicutureMade, which is inevitable
  - SwitchedOn, which is inevitable
  - SwitchedOnFailed, which is inevitable
  - BatteryEmpty, which is optional

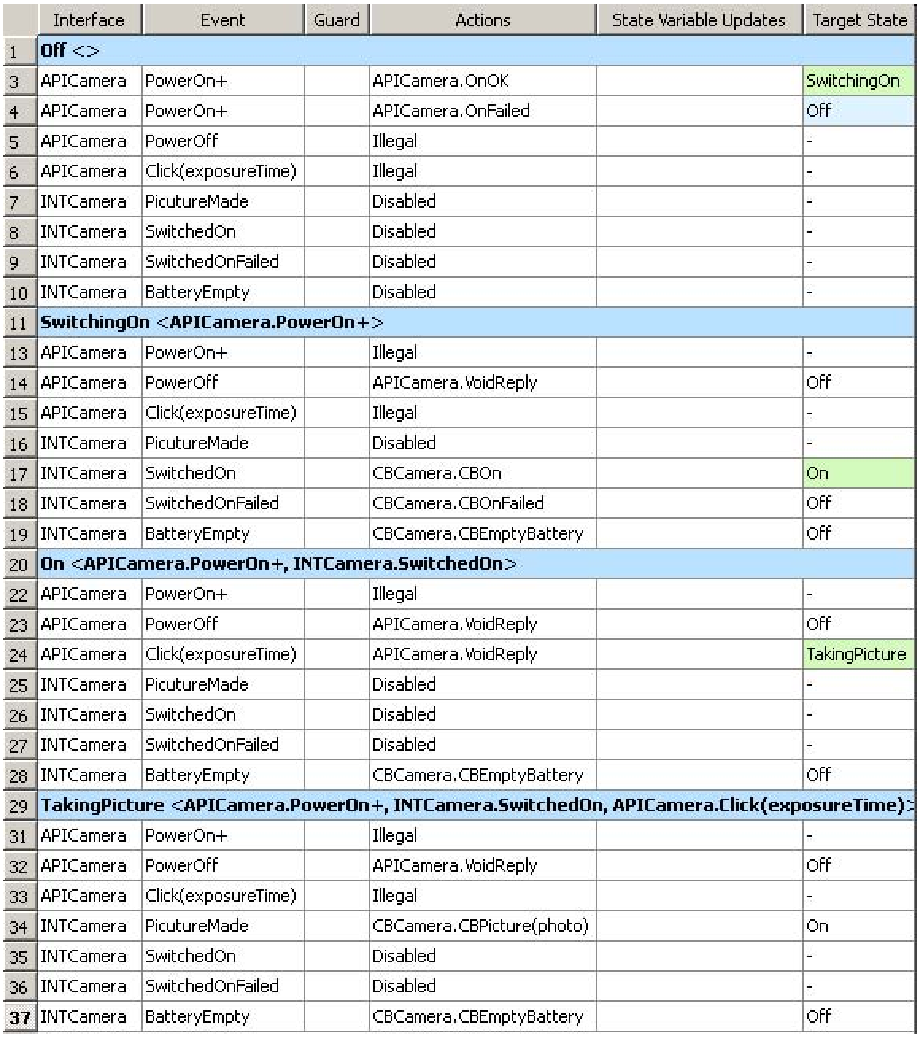
An interface model is represented as a state machine which defines the set of possible traces representing the interface between client and server. Such an ASD interface model plays a similar role as a protocol state machine of UML [2]. An ASD interface not only describes the services offered by the server; it also specifies the calls allowed by the client. So it can be seen as a contract between client and server, similar to the Design by Contract approach [14].

In the Camera example, the ASD interface is shown in Fig. 2. There are four states: Off, SwitchingOn, On, and TakingPicture. In each state the response to all possible stimuli is defined. The ”+” behind the name of an event indicates that it is a valued call. The tool ASD:Suite generates for each reachable state a so-called canonical sequence which is a minimal sequence of input stimuli leading to the state from the initial state (which is always the first state mentioned). This canonical sequence is written behind the name of each state. Observe that the state machine is non-deterministic; in state Off there are two possible responses to the valued call PowerOn.

In state Off the calls PowerOff and Click are declared to be Illegal which means that the client should not call these functions in this state. The modeling events are Disabled in state Off. The use of these internal modeling events is illustrated by state SwitchingOn; when modeling event SwitchedOn or SwitchedOnFailed occurs, the camera sends the corresponding callback to the client.

Note that the rules 2, 12, 21 and 30 are hidden; they can be used for invariants which are not discussed in this paper. Fig. 3 shows a visual representation of the state machine of this interface, which is generated automatically by ASD:Suite from the table of Fig. 2.

To design the camera component, two other components will be used: a battery component and a shutter component. The interfaces of these components are shown in Fig. 4 and Fig. 5, respectively, where illegal and disabled events are hidden. Also these two interfaces are non-deterministic. Observe that the



	Interface	Event	Guard	Actions	State Variable Updates	Target State
1	**Off <>**					
3	APICamera	PowerOn+		APICamera.OnOK		SwitchingOn
4	APICamera	PowerOn+		APICamera.OnFailed		Off
5	APICamera	PowerOff		Illegal		-
6	APICamera	Click(exposureTime)		Illegal		-
7	INTCamera	PicutureMade		Disabled		-
8	INTCamera	SwitchedOn		Disabled		-
9	INTCamera	SwitchedOnFailed		Disabled		-
10	INTCamera	BatteryEmpty		Disabled		-
11	**SwitchingOn <APICamera.PowerOn+>**					
13	APICamera	PowerOn+		Illegal		-
14	APICamera	PowerOff		APICamera.VoidReply		Off
15	APICamera	Click(exposureTime)		Illegal		-
16	INTCamera	PicutureMade		Disabled		-
17	INTCamera	SwitchedOn		CBCamera.CBOn		On
18	INTCamera	SwitchedOnFailed		CBCamera.CBOnFailed		Off
19	INTCamera	BatteryEmpty		CBCamera.CBEmptyBattery		Off
20	**On <APICamera.PowerOn+, INTCamera.SwitchedOn>**					
22	APICamera	PowerOn+		Illegal		-
23	APICamera	PowerOff		APICamera.VoidReply		Off
24	APICamera	Click(exposureTime)		APICamera.VoidReply		TakingPicture
25	INTCamera	PicutureMade		Disabled		-
26	INTCamera	SwitchedOn		Disabled		-
27	INTCamera	SwitchedOnFailed		Disabled		-
28	INTCamera	BatteryEmpty		CBCamera.CBEmptyBattery		Off
29	**TakingPicture <APICamera.PowerOn+, INTCamera.SwitchedOn, APICamera.Click(exposureTime)>**					
31	APICamera	PowerOn+		Illegal		-
32	APICamera	PowerOff		APICamera.VoidReply		Off
33	APICamera	Click(exposureTime)		Illegal		-
34	INTCamera	PicutureMade		CBCamera.CBPicture(photo)		On
35	INTCamera	SwitchedOn		Disabled		-
36	INTCamera	SwitchedOnFailed		Disabled		-
37	INTCamera	BatteryEmpty		CBCamera.CBEmptyBattery		Off

Fig. 2. ICamera: Interface Model of the Camera Component

battery interface uses a state variable EmptyDetected to describe the set of allowed traces. Rule 16 of Fig. 4 shows that a Charge call in state BatteryOn does not lead to an externally visible action, but it updates variable EmptyDetected if it is true. Hidden is the rule which expresses that the Charge call is illegal if EmptyDetected equals false.

These interfaces can be checked using the built-in model checker of ASD:Suite which verifies a number of properties such as guard completeness, absence of state invariant violations, absence of livelock (a livelock occurs when a component is permanently busy with internal behaviour without any visible response to the client), and absence of deadlock (a deadlock occurs when nothing can happen and the component refuses all communication).

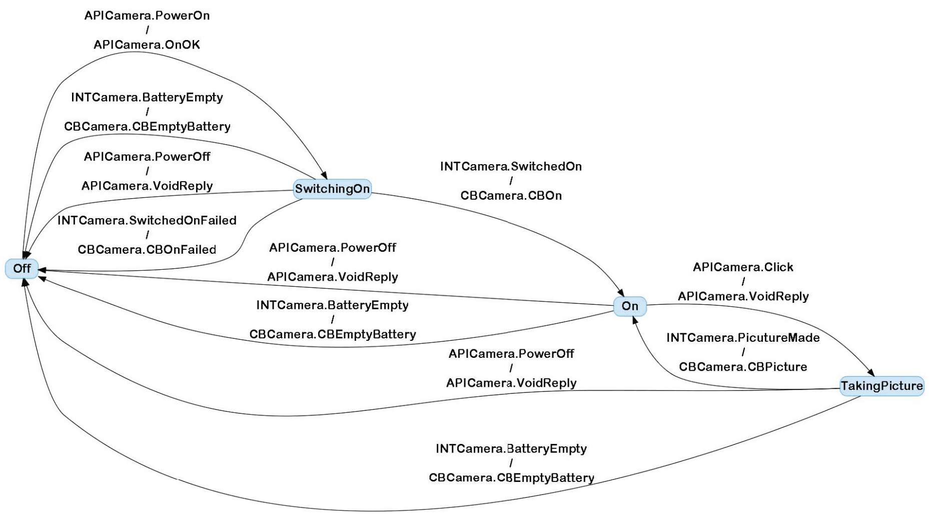
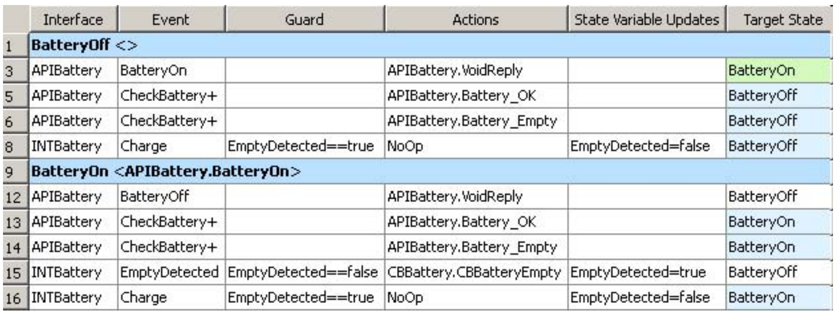
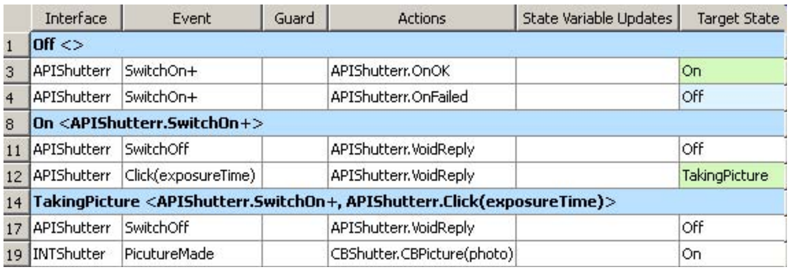

APICamera.PowerOn / APICamera.OnOK
INTCamera.BatteryEmpty / CBCamera.CBEmptyBattery
APICamera.PowerOff / APICamera.VoidReply
SwitchingOn
INTCamera.SwitchedOn / CBCamera.CBOn
INTCamera.SwitchedOnFailed / CBCamera.CBOnFailed
Off
APICamera.PowerOff / APICamera.VoidReply
INTCamera.BatteryEmpty / CBCamera.CBEmptyBattery
On
APICamera.Click / APICamera.VoidReply
INTCamera.PicutureMade / CBCamera.CBPicture
APICamera.PowerOff / APICamera.VoidReply
TakingPicture
INTCamera.BatteryEmpty / CBCamera.CBEmptyBattery

Fig. 3. Graphical Representation of the Interface Model of the Camera



	Interface	Event	Guard	Actions	State Variable Updates	Target State
1	**BatteryOff <>**					
3	APIBattery	BatteryOn		APIBattery.VoidReply		BatteryOn
5	APIBattery	CheckBattery+		APIBattery.Battery_OK		BatteryOff
6	APIBattery	CheckBattery+		APIBattery.Battery_Empty		BatteryOff
8	INTBattery	Charge	EmptyDetected==true	NoOp	EmptyDetected=false	BatteryOff
9	**BatteryOn <APIBattery.BatteryOn>**					
12	APIBattery	BatteryOff		APIBattery.VoidReply		BatteryOff
13	APIBattery	CheckBattery+		APIBattery.Battery_OK		BatteryOn
14	APIBattery	CheckBattery+		APIBattery.Battery_Empty		BatteryOn
15	INTBattery	EmptyDetected	EmptyDetected==false	CBBattery.CBBatteryEmpty	EmptyDetected=true	BatteryOff
16	INTBattery	Charge	EmptyDetected==true	NoOp	EmptyDetected=false	BatteryOn

Fig. 4. IBattery: Interface Model of the Battery Component



	Interface	Event	Guard	Actions	State Variable Updates	Target State
1	**Off <>**					
3	APIShutterr	SwitchOn+		APIShutterr.OnOK		On
4	APIShutterr	SwitchOn+		APIShutterr.OnFailed		Off
8	**On <APIShutterr.SwitchOn+>**					
11	APIShutterr	SwitchOff		APIShutterr.VoidReply		Off
12	APIShutterr	Click(exposureTime)		APIShutterr.VoidReply		TakingPicture
14	**TakingPicture <APIShutterr.SwitchOn+, APIShutterr.Click(exposureTime)>**					
17	APIShutterr	SwitchOff		APIShutterr.VoidReply		Off
19	INTShutter	PicutureMade		CBShutter.CBPicture(photo)		On

Fig. 5. IShutter: Interface Model of the Shutter Component

3 The Usage of Interface Models at Philips Healthcare

The software of the interventional X-ray systems of Philips Healthcare has grown enormously over the last ten years. There is a need to redesign the system architecture to be prepared for a number of challenges:

– Fast support for medical innovations in various clinical segments, such as cardiovascular, neurology, electrophysiology, and surgery.
– Deal with over 1 million product variations, e.g., due to many possible configurations for patient tables, many types of display screens and user input devices, a large number of dedicated devices and image enhancement algorithms supporting specific medical procedures, and combinations of these features.
– Incorporate parts of 3rd party suppliers in a fast and reliable way.
– Allow product service for the next ten years.

To this end, the current system is migrated to a new component-based architecture with clearly defined component interfaces. As a starting point for a new architecture, the system has been split into three main components:

– A *Data Handling* component which deals with patient data, treatment selection, user interaction, interaction with the hospital information system, and the coordination of the other two components.
– An *Image Acquisition* component which deals with the control of the image acquisition chain, including the processing of user input (e.g., via pedals and switches) and the coordination between the X-ray generation (with the right dose for the selected procedure) and the detection of X-ray images.
– An *Image Processing* component which deals with processing and enhancing images, possibly combining it with images from other sources.

The interface between the Image Acquisition and Image Processing components has not been specified in ASD, because this concerns image formats and associated data. The interfaces between Data Handling and the other two components have been defined formally by means of ASD interfaces. These state machines specify an interaction protocol between the components, with several phases such as version exchange, activation & deactivation, data handling, and image acquisition. This leads to large ASD tables. For instance, the interface between Data Handling and Image Acquisition, called *IDHIA*, contains 41 calls, 32 callbacks, and 32 modeling events. The ASD state machine contains 2 state variables and 25 states. In each state the response to the 41 calls and 32 modeling events has to be defined, leading to a table with more than 1800 rule cases.

The three main components of the architecture have been refined and at appropriate places additional ASD interfaces have been defined or are being defined. The general experience is that the formally defined ASD interfaces are very useful. Besides the usual static description, listing all function calls, now also the dynamic behaviour is specified, such as the allowed order of calls and callbacks. Since the interface specifications must be complete, all unclarities have to be resolved and implicit domain knowledge must be made explicit. An important advantage is that the formally defined state machines allow independent development of the components. Moreover, at Philips the ASD interfaces are used to generate conformance tests. Altogether, these formal interfaces reduce the amount of integration problems.

We observed that the definition and interpretation of ASD interface models is sometimes difficult for software developers. It requires the ability to reason about traces and to abstract from design decisions which is not always easy. The difficulties encountered are partly due to the current status of the tool which does not support a very concise notation. An interface model is a flat state machine and structuring mechanisms such as the hierarchy and concurrency constructs of Harel's Statecharts notation [10] are not allowed. Hence a transition which is common to a number of states (e.g., a reset or error transition) has to be entered manually many times. This makes it difficult to read, to understand, and to maintain large interfaces. Current solution is to draw separate Visio diagrams of more structured charts, but it is difficult to keep model and diagram consistent. In any case, it is good practice to keep interfaces small and split them whenever possible.

4 Design Models and Model Checking in ASD:Suite

To implement components, the ASD approach contains so-called design models. Design models are tables similar to interface models with a few differences. A design model must be deterministic and it has a number of associated interface models: an implemented interface model and a number of used interface models.

The model checker of ASD:Suite can be used to check conformance with the implemented interface and consistency with the used interfaces. Complete executable code can be generated from the design model, where a choice can be made between a number of programming languages (currently C, C++, C#, and Java). The camera example is used to explain the ASD design models in Sect. 4.1 and the model checker in Sect. 4.2.

4.1 ASD Design Models

The design of the camera component is depicted in Fig. 6, where ovals represent ASD interface models and the rectangle denotes an ASD design model. The up arrow denotes that the design model refines the interface model of Fig. 2, as will be explained in Sect. 4.2. The design uses the interfaces IBattery of the battery (Fig. 4) and IShutter of the shutter (Fig. 5).

Figure 7 shows the ASD design model for the camera example, hiding illegal and disabled events. Similar to the interface model, the table must be complete in the sense that for all events an action must be defined (which might be Illegal or Disabled). The events now include callbacks of the used interfaces, such as the callbacks CBBatteryEmpty from the battery and CBPicture from the shutter. Similarly, the actions may contain calls to used interfaces. For instance, in state

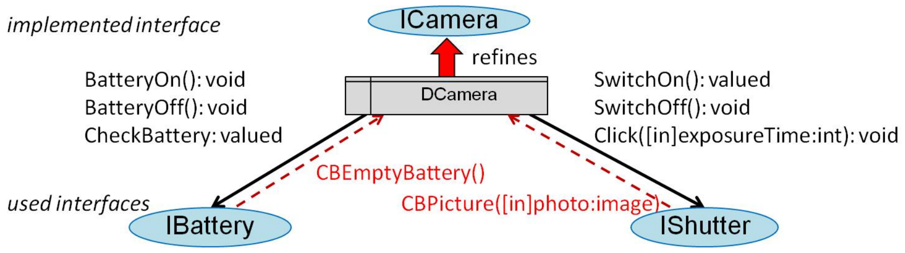
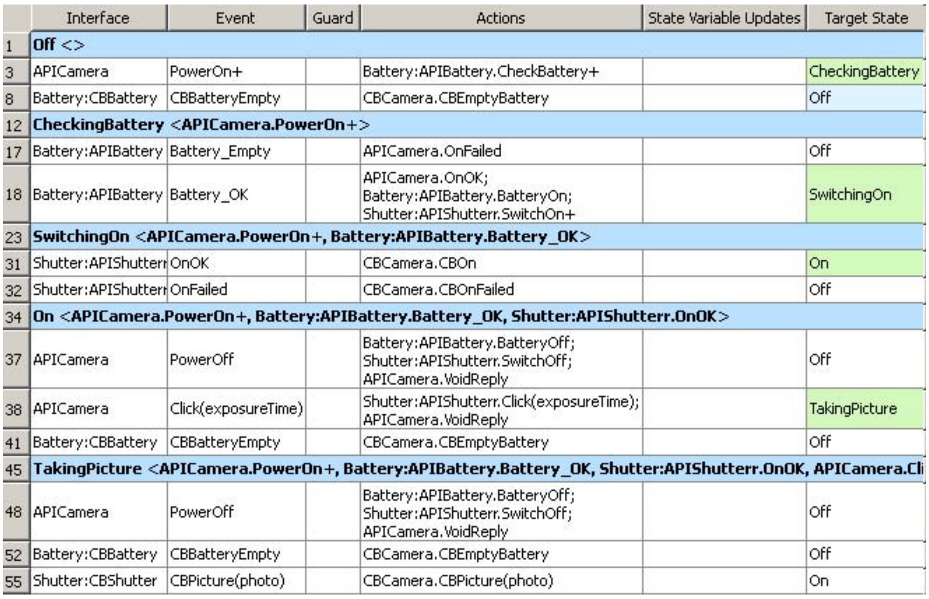


Fig. 6. Design of the Camera Component

Off rule 3 expresses that - as a response to the PowerOn call - the function CheckBattery of the battery is called. As indicated by the "+" behind the name, this is a valued call and in state CheckingBattery the component is waiting for a response. In such a so-called blocking state, all other events are blocked.



	Interface	Event	Guard	Actions	State Variable Updates	Target State
1	**Off <>**					
3	APICamera	PowerOn+		Battery:APIBattery.CheckBattery+		CheckingBattery
8	Battery:CBBattery	CBBatteryEmpty		CBCamera.CBEmptyBattery		Off
12	**CheckingBattery <APICamera.PowerOn+>**					
17	Battery:APIBattery	Battery_Empty		APICamera.OnFailed		Off
18	Battery:APIBattery	Battery_OK		APICamera.OnOK; Battery:APIBattery.BatteryOn; Shutter:APIShutterr.SwitchOn+		SwitchingOn
23	**SwitchingOn <APICamera.PowerOn+, Battery:APIBattery.Battery_OK>**					
31	Shutter:APIShutterr	OnOK		CBCamera.CBOn		On
32	Shutter:APIShutterr	OnFailed		CBCamera.CBOnFailed		Off
34	**On <APICamera.PowerOn+, Battery:APIBattery.Battery_OK, Shutter:APIShutterr.OnOK>**					
37	APICamera	PowerOff		Battery:APIBattery.BatteryOff; Shutter:APIShutterr.SwitchOff; APICamera.VoidReply		Off
38	APICamera	Click(exposureTime)		Shutter:APIShutterr.Click(exposureTime); APICamera.VoidReply		TakingPicture
41	Battery:CBBattery	CBBatteryEmpty		CBCamera.CBEmptyBattery		Off
45	**TakingPicture <APICamera.PowerOn+, Battery:APIBattery.Battery_OK, Shutter:APIShutterr.OnOK, APICamera.Cli**					
48	APICamera	PowerOff		Battery:APIBattery.BatteryOff; Shutter:APIShutterr.SwitchOff; APICamera.VoidReply		Off
52	Battery:CBBattery	CBBatteryEmpty		CBCamera.CBEmptyBattery		Off
55	Shutter:CBShutter	CBPicture(photo)		CBCamera.CBPicture(photo)		On

Fig. 7. DCamera: Design Model of the Camera Component

The design model assumes that CBPicture only occurs in state TakingPicture; in all other states the callback is illegal or blocked. The correctness of this assumption is verified by the model checker using the interface specification of the shutter. CBPicture is a so-called *solicited* callback, because it is received as a response to the Click call to the shutter. On the other hand, CBBatteryEmpty can be received in any non-blocking state and is called an *unsolicited* callback.

In an ASD design model it is not possible to make control decisions based on parameters of function calls. For instance, in the camera example it is not possible to make a case distinction on the exposureTime parameter of the Click call. If control would depend on the value of such a parameter, it has to be sent to a foreign component, which is not implemented using ASD. Such a foreign component can analyze the value and return different values or callbacks to indicate the required control.

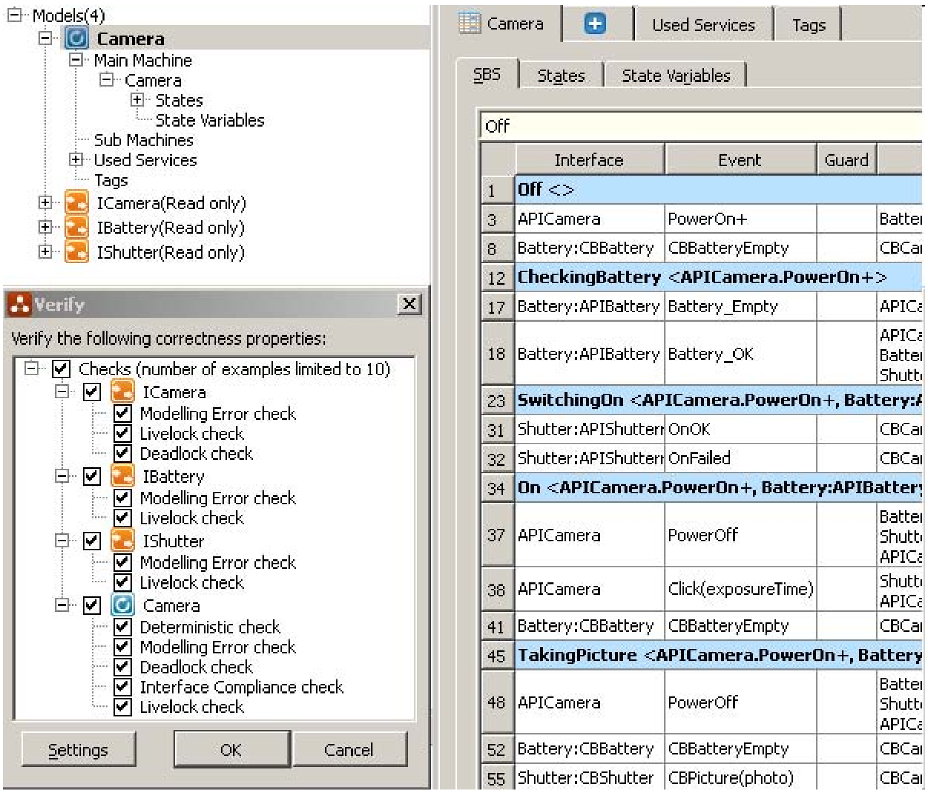
The semantics of a design model is such that callbacks from used components can always be received and they are put into a so-called *callback queue* (FIFO). Client calls are serialized, that is, at any point in time at most one client call is executed. Callbacks have priority over client calls. Initially, and after the completion of a rule case, first the callback queue is inspected. If this queue is not empty, the rule case corresponding to the first callback is executed. When the callback queue is empty and no call is being processed, a new call may be accepted. An illegal call or callback leads to a halt of the component.

4.2 Compositional Model Checking

The tool ASD:Suite contains a fixed number of checks on design models. Figure 8 shows a screenshot of part of the tool with a Verify window. Verification includes the previously discussed checks on all interfaces, i.e., implemented and used interfaces. In addition there are specific checks for design models, such as a check to ensure that the design model is deterministic. Most important is a check on the consistency of all interfaces. The design should adhere to all interface models of used components and it should conform to the implemented interface. Conformance has been defined formally in the failures-divergence model of CSP [17] and is checked with the underlying FDR2 model checker [8]. Note that FDR2, which is an abbreviation of Failures/Divergence Refinement 2, is in fact a refinement checker.

In the camera example, verification revealed quite a number of problems in the models presented above. A few errors found by the model checker:

– In used interface IBattery it is possible to get a BatteryOff call in state BatteryOff; this is a race condition between a PowerOff call and a callback CBBatteryEmpty send by the battery components. Note that the callback is put into the callback queue of the camera component while the BatteryOff call is processed. This problem is corrected by improving the battery interface as shown in Fig. 9 where rule 4 now allows a BatteryOff call.
– The model checker complained about an attempt to switch the shutter on when it was already on, which is not allowed by the interface of the shutter as specified in Fig 5. Analyzing this situation, it turned out that in the design model it was forgotten to switch the shutter off when a CBBatteryEmpty has been received (rules 41 and 52 in Fig. 7).
– Similarly, it was forgotten to switch the battery off when an attempt to switch the shutter on fails (rule 32 in Fig. 7).
– As another race condition, the model checker shows that a callback CBPicture might be received in state Off, namely after a CBBatteryEmpty in state TakingPicture. This has been repaired by adding a rule case in the design to receive the callback, but not forwarding it to the client.



```
Models(4)
  Camera
    Main Machine
      Camera
        States
        State Variables
    Sub Machines
    Used Services
    Tags
  ICamera(Read only)
  IBattery(Read only)
  IShutter(Read only)
```

Verify

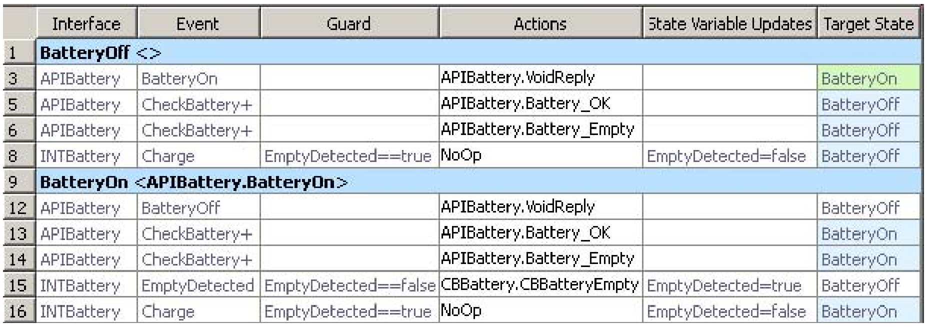
Verify the following correctness properties:

```
Checks (number of examples limited to 10)
  ICamera
    Modelling Error check
    Livelock check
    Deadlock check
  IBattery
    Modelling Error check
    Livelock check
  IShutter
    Modelling Error check
    Livelock check
  Camera
    Deterministic check
    Modelling Error check
    Deadlock check
    Interface Compliance check
    Livelock check
```

Off

	Interface	Event	Guard	
1	**Off <>**			
3	APICamera	PowerOn+		Batte
8	Battery:CBBattery	CBBatteryEmpty		CBCa
12	**CheckingBattery <APICamera.PowerOn+>**			
17	Battery:APIBattery	Battery_Empty		APICa
18	Battery:APIBattery	Battery_OK		APICa Batte Shutt
23	**SwitchingOn <APICamera.PowerOn+, Battery:A**			
31	Shutter:APIShutter	OnOK		CBCa
32	Shutter:APIShutter	OnFailed		CBCa
34	**On <APICamera.PowerOn+, Battery:APIBatter**			
37	APICamera	PowerOff		Batte Shutt APICa
38	APICamera	Click(exposureTime)		Shutt APICa
41	Battery:CBBattery	CBBatteryEmpty		CBCa
45	**TakingPicture <APICamera.PowerOn+, Battery**			
48	APICamera	PowerOff		Batte Shutt APICa
52	Battery:CBBattery	CBBatteryEmpty		CBCa
55	Shutter:CBShutter	CBPicture(photo)		CBCa

Fig. 8. Screenshot ASD:Suite with Verify Window



	Interface	Event	Guard	Actions	State Variable Updates	Target State
1	**BatteryOff <>**					
3	APIBattery	BatteryOn		APIBattery.VoidReply		BatteryOn
5	APIBattery	CheckBattery+		APIBattery.Battery_OK		BatteryOff
6	APIBattery	CheckBattery+		APIBattery.Battery_Empty		BatteryOff
8	INTBattery	Charge	EmptyDetected==true	NoOp	EmptyDetected=false	BatteryOff
9	**BatteryOn <APIBattery.BatteryOn>**					
12	APIBattery	BatteryOff		APIBattery.VoidReply		BatteryOff
13	APIBattery	CheckBattery+		APIBattery.Battery_OK		BatteryOn
14	APIBattery	CheckBattery+		APIBattery.Battery_Empty		BatteryOn
15	INTBattery	EmptyDetected	EmptyDetected==false	CBBattery.CBBatteryEmpty	EmptyDetected=true	BatteryOff
16	INTBattery	Charge	EmptyDetected==true	NoOp	EmptyDetected=false	BatteryOn

Fig. 9. Improved Battery Interface IBattery

ASD:Suite has a nice visualization of error traces, which makes it easy to find the cause of an error. Figure 10 shows the visualization of the last problem mentioned above. It shows the lifeline of the Camera component, with a client, the callback queue (called DPC+Q), its used components Battery and Shutter, and an environment which triggers modeling events in the used interfaces.

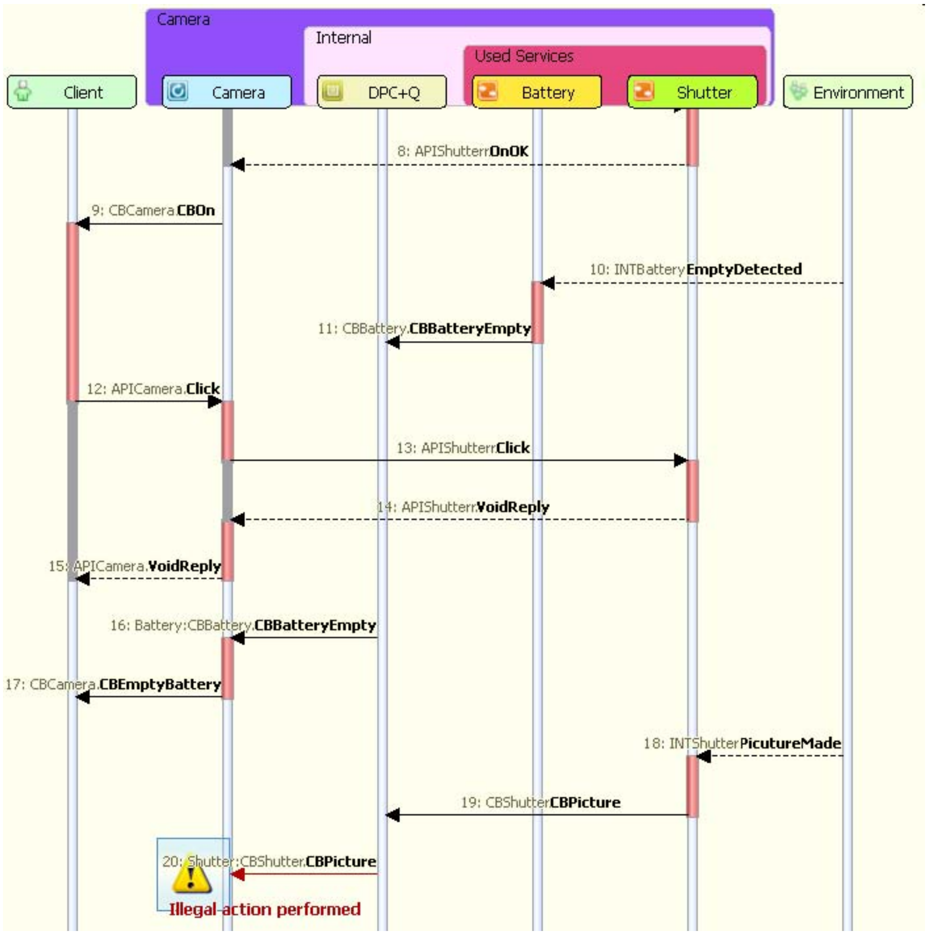
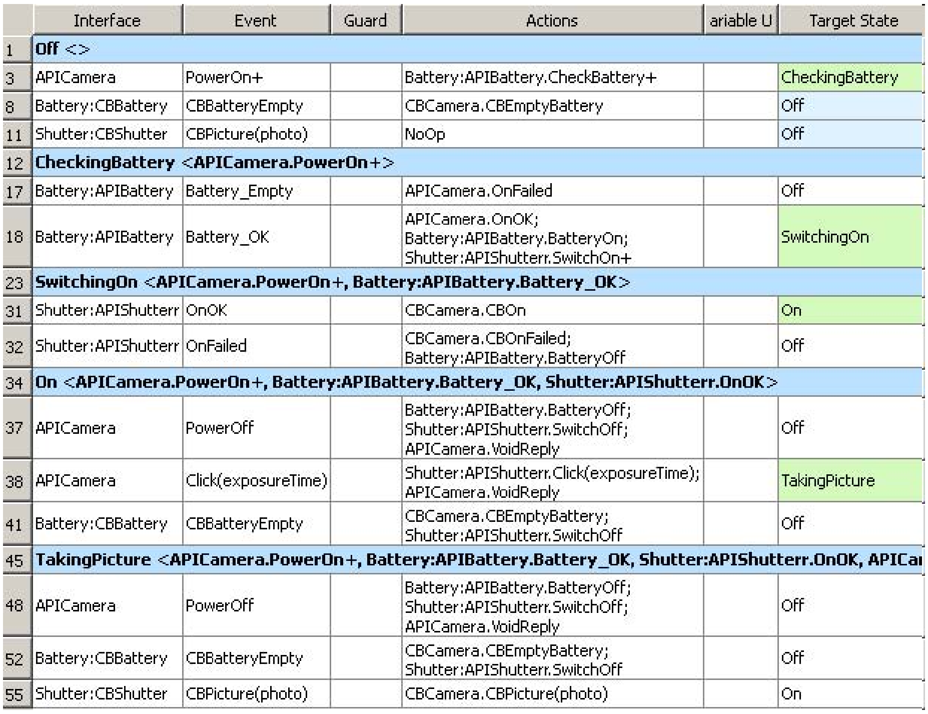


Fig. 10. Visualization of an Error Trace

The corrected design model for the camera is shown in Fig. 11, with changes in rules 11, 32, 41, and 52.

Observe that the verification is compositional since it uses only the interfaces of the used components. Hence, the used components can be developed independently according to their interface. Also note that there is no obligation to develop these components with ASD. Typically, components that involve data manipulations will be implemented differently and conformance to their ASD interface is checked by means of testing.

Finally, observe that the components have a strict communication pattern; a client of a component can only perform synchronous calls and might receive asynchronous callbacks from the component. Similarly, the component itself will only perform synchronous calls on its used components and receive callbacks from them. In this way absence of deadlock is achieved by construction.



	Interface	Event	Guard	Actions	ariable U	Target State
1	**Off <>**					
3	APICamera	PowerOn+		Battery:APIBattery.CheckBattery+		CheckingBattery
8	Battery:CBBattery	CBBatteryEmpty		CBCamera.CBEmptyBattery		Off
11	Shutter:CBShutter	CBPicture(photo)		NoOp		Off
12	**CheckingBattery <APICamera.PowerOn+>**					
17	Battery:APIBattery	Battery_Empty		APICamera.OnFailed		Off

18	Battery:APIBattery	Battery_OK		APICamera.OnOK;
				Battery:APIBattery.BatteryOn;
				Shutter:APIShutterr.SwitchOn+
						SwitchingOn

23	**SwitchingOn <APICamera.PowerOn+, Battery:APIBattery.Battery_OK>**					
31	Shutter:APIShutterr	OnOK		CBCamera.CBOn		On
32	Shutter:APIShutterr	OnFailed		CBCamera.CBOnFailed; Battery:APIBattery.BatteryOff		Off
34	**On <APICamera.PowerOn+, Battery:APIBattery.Battery_OK, Shutter:APIShutterr.OnOK>**					

37	APICamera	PowerOff		Battery:APIBattery.BatteryOff;
				Shutter:APIShutterr.SwitchOff;
				APICamera.VoidReply
						Off

38	APICamera	Click(exposureTime)		Shutter:APIShutterr.Click(exposureTime); APICamera.VoidReply		TakingPicture
41	Battery:CBBattery	CBBatteryEmpty		CBCamera.CBEmptyBattery; Shutter:APIShutterr.SwitchOff		Off
45	**TakingPicture <APICamera.PowerOn+, Battery:APIBattery.Battery_OK, Shutter:APIShutterr.OnOK, APICa**					

48	APICamera	PowerOff		Battery:APIBattery.BatteryOff;
				Shutter:APIShutterr.SwitchOff;
				APICamera.VoidReply
						Off

52	Battery:CBBattery	CBBatteryEmpty		CBCamera.CBEmptyBattery; Shutter:APIShutterr.SwitchOff		Off
55	Shutter:CBShutter	CBPicture(photo)		CBCamera.CBPicture(photo)		On

Fig. 11. Correct Design Model for the Camera

5 The Use of Design Models at Philips Healthcare

In this section we describe experiences with the use of ASD design models at Philips Healthcare. In Sect. 5.1 we report about the attempts to design a component that has to satisfy a large and complex ASD interface. Next Sect. 5.2 summarizes the observations made when applying the ASD approach to the detailed design of components.

5.1 Decomposition of a Complex Interface

At Philips Healthcare, ASD-based design has been applied to components which are responsible for high-level supervisory control. Starting point are the ASD interface models between the main system components as mentioned in Sect. 3. In this section we describe the experiences with ASD design in the Image Acquisition component. Hence, we start with the complex interface model IDHIA, where Data Handling is the client and Image Acquisition the server. First challenge was to design a component which conforms to this complex interface. We describe a few design possibilities considered and list the resulting observations.

The main idea of the design was to start with a component which decomposes the interface into a number of smaller interfaces, based on the phases defined in the interface protocol: version exchange, activation/deactivation, data handling, and image acquisition. We investigated two possibilities:

– A sequence of components, where the first one (called DVersion) manages the version part of the protocol, using an interface IVersion for detailed control, and forwarding the remainder of the interface to a reduced interface. The second one deals with activation, and the third one with the data handling and image acquisition part. The main idea of the design is depicted in Fig. 12, using the same notation as Fig. 6.

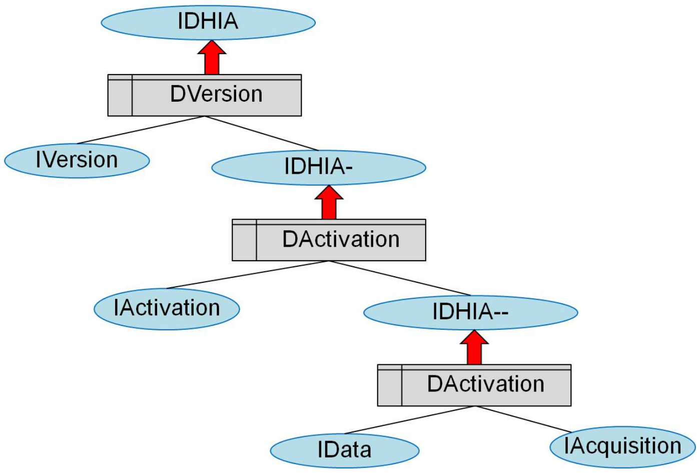


Fig. 12. Design with Sequence of Components

– A single component which has as states the main phases of the protocol (Version, Activation, Data and Acquisition). In each state it forwards relevant calls to the corresponding interface. This design is depicted in Fig. 13.

When trying to implement these design possibilities in ASD, we encountered a number of problems. Main problem was that calls that are illegal in the interface cannot be accepted and forwarded. To avoid these problems, all calls can be made valued (instead of void), returning Accepted or Rejected. But this was not acceptable for the designers of the Data Handling component, because valued calls require an additional state to wait for the result and too many of them lead to a complex design.

The problems with illegal calls also triggered a discussion about robustness. As mentioned in Sect. 4.1, the semantics is such that any illegal client call leads to a halt of the component. To be more robust against illegal calls of non-ASD clients (e.g., 3rd party clients) Verum was requested to allow the possibility to specify some kind of graceful degradation for illegal calls of non-ASD clients. To deal with illegal callbacks of non-ASD used components, a pattern has been

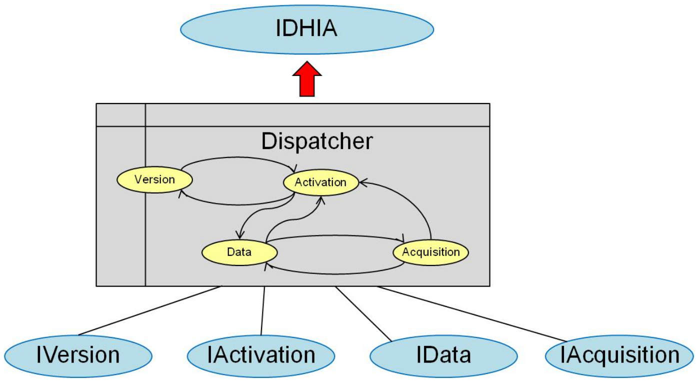


Fig. 13. Design with Dispatcher Component

defined by Verum to deal with such callbacks and to log erroneous callbacks (e.g., to discuss interface errors with suppliers of 3rd party used components).

Finally, we observed that forwarding calls leads to additional race conditions. A race condition shows a design problem to deal with a simultaneous client call and a callback from a used interface where the order of acceptance leads to different states. A pattern has been defined to solve race conditions in a design systematically, but altogether both design options discussed above did not lead to satisfactory implementations in ASD. Verum is working on alternative verification modes where illegal calls can be delegated.

Meanwhile, the choice was made to design a component which contains all (25) states of the interface to be able to decide which calls are illegal. All non-illegal calls are forwarded to appropriate interfaces, similar to the designs pro-posed above. Compared to the solution where all calls are made valued, this leads to easier interfaces. It also avoids additional race conditions. The design, however, is rather complex and not easy to read and to maintain. A small improvement could be achieved by introducing sub-state machines which can be used one-level deep in design models (but not in interface models). This last step, however, required an improvement of interface model IDHIA where the phases of the communication protocol had to be separated more clearly.

5.2 Detailed Design of Control Components

Next we describe experiences with the detailed design of the high-level control part of the Image Acquisition component. To be able to implement the main control components in ASD, a number of constraints have to be taken into account:

– The communication pattern with synchronous function calls from top to bottom and callbacks from bottom to top has to be respected
– Data manipulation and control has to be separated to allow the implementation of the data part in non-ASD components

– To allow model checking and to avoid hitting the state explosion problem:
  - Reduce the number of callbacks that may occur simultaneously
  - Reduce the number of unsolicited callbacks
  - Reduce the number of state variables
  - Keep components small, split components when adding functionality

Although the compositional approach of ASD is very important for scalability and allows the application to large industrial systems, typically the size of some components increased too much during development, especially when exceptions and failure modes were added. Then easily the boundaries of the model checker were met and a number of redesigns were needed to make model checking possible again.

In general, it turned out to be very important to have an extensive preparation phase before applying ASD. The requirements must be clarified with more precision than during conventional software development and extensive discussions on both requirements and design are needed. Since ASD:Suite considers each component and its interface in isolation, additional tooling (including a UML tool, PowerPoint and Visio) has been used to create design overviews.

Once the requirements were clarified and a proper design with small components was established, we usually started with a precise definition of component interfaces in ASD. Next the completion of design models was rather straightforward. Of course, the model checker still finds quite a number of errors, but they are often rather easy to correct. The camera example is a good illustration of the type of errors found and their correction.

6 Concluding Remarks

Summarizing the experiences with the formal development tool ASD:Suite at Philips Healthcare, it is clear that the formally defined interfaces are very beneficial. They not only describe the syntax of calls and callbacks, but also define the expected behaviour in terms of the allowed sequences of events. This reduces the amount of faults detected during integration and improves the possibility to develop components independently. At Philips Healthcare the aim is to certify the main components using conformance tests which are derived from the ASD interface models. Main challenge is to keep the interfaces manageable and maintainable.

The most important advantage of ASD-based design is the strong combination of compositional model checking and complete code generation from the same models, where code and model have the same semantics. By means of the model checker many faults are found early and the nice visualization of error traces makes it easy to correct them. Especially when dealing with complex combinations of device failures, activation/deactivation, and connect/disconnect behaviour it would be almost impossible to obtain a correct implementation without the support of a tool like ASD:Suite.

An analysis of earlier applications of ASD in the Data Handling component at Philips Healthcare [9] showed that units containing ASD components have less reported defects than other units. Although the work on the Image Acquisition part was not yet completed at the time of writing, the impression is that the problems reported after the integration and test phase of the first increments mainly concerned differences in the interpretation of requirements.

It is important to realize that a number of design constraints have to be fulfilled in order to apply ASD successfully, such as the required communication pattern. At Philips Healthcare, the hierarchical control structure required by ASD supports the desired change from an object-oriented blackboard architecture towards a hierarchical component-based control architecture. On the other hand, during detailed design it is sometimes difficult to make a balanced division into ASD components for the control part and non-ASD components to deal with data manipulations. Moreover, components have to be made smaller than usual to allow model checking.

Another important observation is that the engineers have to realize that testing is still needed and a good set of unit tests is still very valuable. The ASD model checker verifies only the interaction protocol between components; it does not check the relation between calls to the implemented interface and calls to the used interfaces. It would be an interesting challenge to investigate whether the specification and verification of such relations could be added to the tool suite without compromising the ease of use for the average engineer.

Acknowledgments. This work has been supported by the Dutch knowledge workers project Partitura, ITEA project Care4Me, and COMMIT project Allegio. We would like to thank Leon Bouwmeester of Verum for his valuable comments on a draft of this paper.

References

1. Abrial, J.R.: The B-book: assigning programs to meanings. Cambridge University Press, New York (1996)
2. Booch, G., Rumbaugh, J.E., Jacobson, I.: The unified modeling language user guide - the ultimate tutorial to the UML from the original designers. Addison-Wesley object technology series. Addison-Wesley-Longman (1999)
3. Broadfoot, G.H., Broadfoot, P.J.: Academia and industry meet: Some experiences of formal methods in practice. In: 10th Asia-Pacific Software Engineering Conferenc (APSEC 2003), pp. 49–58 (2003)
4. ClearSy: Atelier B, industrial tool supporting the B method (2011), http://www.atelierb.eu/en/
5. CSK Systems Corporation: VDMTools, industrial tool supporting VDM++ (2011), http://www.vdmtools.jp/en/
6. Esterel Technologies: SCADE Suite, model based development environment dedicated to critical embedded software (2011), http://www.esterel-technologies.com/products/scade-suite/
7. Fitzgerald, J., Larsen, P.G., Mukherjee, P., Plat, N., Verhoef, M.: Validated Designs for Object-oriented Systems. Springer, New York (2005), http://www.vdmbook.com
8. Formal Systems (Europe) Ltd: FDR2 model checker (2011), http://www.fsel.com/

9. Groote, J.F., Osaiweran, A., Wesselius, J.H.: Analyzing the effects of formal methods on the development of industrial control software. In: Proceedings of the IEEE ICSM 2011, pp. 467–472 (2011)
10. Harel, D.: Statecharts: A visual formalism for complex systems. Science of Computer Programming 8(3), 231–274 (1987)
11. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall (1985)
12. Hooman, J.: Specification and Compositional Verification of Real-Time Systems. LNCS, vol. 558. Springer, Heidelberg (1991)
13. Hopcroft, P.J., Broadfoot, G.H.: Combining the box structure development method and CSP for software development. Electronic Notes in Theoretical Computer Science 128(6), 127–144 (2005)
14. Meyer, B.: Applying "design by contract". IEEE Computer 25(10), 40–51 (1992)
15. Mills, H.D.: Stepwise refinement and verification in box-structured systems. Computer 21, 23–36 (1988)
16. Prowell, S.J., Poore, J.H.: Foundations of sequence-based software specification. IEEE Transactions on Software Engineering 29, 417–429 (2003)
17. Roscoe, A.W.: The theory and practice of concurrency. Prentice Hall (1998)
18. Verum: ASD:Suite (2011), http://www.verum.com/

Model Checking Workflow Monitors and Its Application to a Pain Management Process

Fazle Rabbi, Ahmed Shah Mashiyat, and Wendy MacCaull

Centre for Logic and Information
St. Francis Xavier University
{rfazle,x2008ooc,wmaccaul}@stfx.ca

Abstract. Safety critical systems often consist of many complex processes. Monitoring the behaviour of such processes is critical for enforcing policies, and achieving efficiency and reliability goals. The paper presents a new graphical language for modeling monitors for time constrained workflow processes. To ensure the correctness of a monitor system, we present a model checking approach for verification. We provide an automated data aware translation from a monitor model and its associated workflow to DVE, the input language of the DiVinE model checker, reducing the time and effort required for model checking. We prove the correctness of the translation and give a detailed case study involving LTL properties for a compensable healthcare workflow with time constraints and monitors.

1 Introduction

Monitoring the real time behaviour of safety critical systems and providing alerts when system specifications are not met is routinely done. For example, in healthcare, a safety critical system, the status of a patient in an intensive care unit is constantly monitored. But one can also see that the timely availability of information and the coordination of work are of extreme importance in many processes that require the cooperation of many different organizational units, professional specialties, and geographical locations. For healthcare errors, miscommunications, and gaps in care are costly for the system, may adversely affect the patient, and may lead to the patient's death. Implementing a workflow process to orchestrate the care can overcome some of these problems, as the control flow can ensure correct process. However, the nature of healthcare, which is highly variable due to the physiology of patients and other circumstances, some of which depend on the healthcare setting, necessitates that healthcare providers frequently override normal processes to deal with unforeseen situations and crises in real time. Monitoring is therefore especially critical for healthcare processes, and integrating a monitoring mechanism with an existing healthcare Workflow Management System (WfMS) can greatly increase the effectiveness of the system.

It is important to verify that a workflow together with an associated monitor mechanism works as expected, before enactment. Applying formal methods, e.g., model checking, can help to ensure its correctness. Workflow and an associated monitor for safety critical system are often time sensitive. For instance, in an emergency department, after an emergency case arrives at the hospital, standard model checking can only verify whether for a particular process *"Eventually the patient receives a certain treatment"*, but to save the patient's life, it should be verified that the process satisfies *"The patient receives a certain treatment within a reasonable time"*. Typically monitor systems are configured to notify clinicians about any abnormal situation in the process. Rollback of flow and associated compensation of side effects are often required as care is modified to suit patient needs. The model for a monitor for a time constrained compensable workflow may be complex and its behaviour must be verified before their start of operation.

In this paper, we present a modular time constrained workflow monitor (called WMon-net) and its graphical modeling language which is based on time Petri nets [4]. We show how to integrate such a monitor with workflow models; using the Timed Compensable Workflow Modeling Language (CWML$_T$) [13]. Using compensable tasks allows us to design fault tolerant systems. We extend our workflow management system, NOVA Workflow[1], to allow the user to graphically input a timed compensable workflow together with workflow monitors and a specification in LTL and automatically verify if the specification holds, thus providing a method to verify a model of a workflow monitor along with its associated workflow, using an LTL-based model checker. We give a translation method to automatically generate a model for the workflow and its monitor in DVE, the input language of parallel distributed model checker DiVinE [1], and provide a proof of correctness of the translation. Since DiVinE is an untimed model checker, we use Lamport's [10] explicit time description method (where time is simulated as states) to model and verify the time information. NOVA Workflow is built using Service Oriented Architecture (SOA) and generates Java code from a workflow model. The NOVA Engine (a component of NOVA Workflow) executes the tasks, manages the control flow, and triggers the monitor when required; the implementation details along with the communication among different components of the system (e.g., workflow tasks, monitors, client applications) may be found in [12]. It is anticipated that the workflow monitor may be integrated with other workflow languages with Petri net based foundations. We used the DiVinE model checker as it has been implemented as a distributed model checker and thus has more memory which is needed for verification of large models.

The rest of the paper is organized as follows. Section 2 gives some preliminaries of CWML$_T$ and its underlying time Petri net based foundation, section 3 presents the graphical language for workflow monitoring, details an automated translation method for the verification of workflow and its monitors, and provides a proof of correctness. A case study is presented in section 4, related works are presented in section 5, and section 6 concludes the paper.

[1] `http://logic.stfx.ca/software/nova-workflow`

2 Preliminaries

An extended model of ACID (Atomic, Consistent, Isolated, Durable) transactions, called *compensable transactions*, can support handling "abnormal" behavior of workflows. A compensable transaction is a type of transaction whose effect can be semantically undone even after it has committed. Li et al. [11] defined the behavioral characteristics of the transaction calculus (t-calculus) operators focusing on compensable transactions. In [16] we defined a Compensable Workflow Modeling Language (CWML) using t-calculus operators. In addition to the t-calculus operators (sequential composition (;), parallel composition ($||$), internal choice ($\sqcap$), speculative choice ($\otimes$), and alternative choice ($\rightsquigarrow$), see Definition. 2), CWML has the basic "sequence" ($\bullet$), "and" ($\wedge$), "xor" ($\times$), "or" ($\vee$) and "loop" ($^+$) operators, all of which are used to compose tasks. In [13] we extended this work with time and proposed the Timed Compensable Workflow Modeling Language (CWML$_T$). Atomic tasks in CWML$_T$ are of two types, i.e., uncompensable and compensable. An atomic timed uncompensable task is an activity which always finishes successfully within the assigned time, if activated. In the case of an error executing the forward flow, an atomic timed compensable task aborts and performs some compensation following the time constraints. The Petri net representations of atomic timed uncompensable and atomic timed compensable tasks are given in Fig. 1 where small rectangles represent transitions and circles represent places. The Petri net representations of compound tasks may be found in [16].

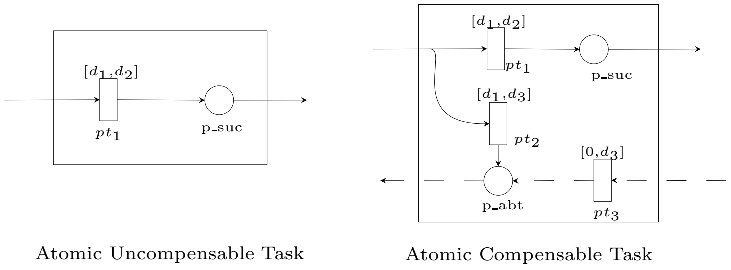


Fig. 1. Petri net representation of atomic tasks

In Fig. 1 solid arcs represent a forward flow and dotted arcs represent a compensation flow; d_1, d_2 and d_3 are the *delay*, *duration*, and *compensation duration* respectively. *Delay* is the time duration between two subsequent activities (i.e., tasks). *Duration* is the maximum time required to finish a task. *Compensation duration* refers to the maximum time required to compensate a failed task plus the *duration* for the task. d_1, d_2 and d_3 are expressed by integer values following the Gregorian calendar i.e., year, month, week, day, hour, and minute.

When a task executes, it performs some actions, and the execution of a task may depend on some conditions; these are defined below:

Definition 1. *A term, σ, is defined using BNF as: $\sigma ::= c \mid \chi \mid \sigma \oplus \sigma$, where $\oplus \in \{+, -, \times, \div\}$, c is a natural number and χ is a (natural) variable. A pre-condition is a formula, ψ^p, is defined as $\psi^p ::= \sigma \diamond \sigma \mid (\psi^p \uplus \psi^p)$, where $\diamond \in \{<, \leq, >, \geq, ==\}$, $\uplus \in \{\&\&, ||\}$ and σ is a term. An action, ψ^a, is an assignment defined as $\psi^a ::= v = \sigma$; v is called a mapsTo variable and σ is a term.*

A compensable task can be composed with other compensable tasks using the t-calculus operators.

Definition 2. *A compensable task, Γ_c, is recursively defined using BNF as: $\Gamma_c ::= \tau_c \mid (\Gamma_{c_1} \odot \Gamma_{c_2})$, where τ_c is an atomic compensable task, which has a set of pre-conditions $\{\psi_i^p\}$ and sets of actions $\{\psi^a\}$ (forward) and $\{\psi^{a'}\}$ (compensation) associated to it, and $\odot \in \{;, ||, \sqcap, \otimes, \rightsquigarrow\}$ is a t-calculus operator defined as follows:*

- *Γ_{c_1} ; Γ_{c_2}: Γ_{c_2} will be activated after the successful completion of Γ_{c_1};*
- *$\Gamma_{c_1} \mid\mid \Gamma_{c_2}$: Γ_{c_1} and Γ_{c_2} will be executed in parallel. If either of them (Γ_{c_1} or Γ_{c_2}) is aborted, the other one will also be aborted;*
- *$\Gamma_{c_1} \sqcap \Gamma_{c_2}$: either Γ_{c_1} or Γ_{c_2} will be activated depending on some internal choice;*
- *$\Gamma_{c_1} \otimes \Gamma_{c_2}$: Γ_{c_1} and Γ_{c_2} will be executed in parallel. The first task that reaches the goal will be accepted and the other one will be aborted;*
- *$\Gamma_{c_1} \rightsquigarrow \Gamma_{c_2}$: Γ_{c_1} will be activated first to achieve the goal, if Γ_{c_1} is aborted, Γ_{c_2} will be executed to achieve the goal.*

Note that, in this paper, we assume if activated, an atomic compensable task τ_c either completes successfully or fully compensates. Therefore, the backward handling operator ($\trianglerighteq$), forward handling operator ($\triangleright$) and programmable compensation operator ($\circledast$) from [16] are omitted. Any task can be composed with uncompensable and/or compensable tasks to create a new task. As above, a task may be considered as a formula; subtasks correspond to subformulas.

Definition 3. *A task, Γ, is recursively defined using BNF as: $\Gamma ::= \tau \mid \Gamma_c \mid (\Gamma_1 \ominus \Gamma_2) \mid (\Gamma_1)^+$, where τ is an uncompensable atomic task, which has a set of pre-conditions $\{\psi^p\}$ and a set of actions $\{\psi^a\}$ associated to it; Γ_c is a compensable task; $\ominus \in \{\wedge, \vee, \times, \bullet\}$ is a binary operator. $^+$ is a unary operator for loops (iteration), and Γ^+ denotes that Γ executes at least once if activated and iterates as long as its pre-conditions are true.*

Any task which is built up from the operators $\{\wedge, \vee, \times, \bullet\}$ is deemed as uncompensable. Thus if Γ_1 and Γ_2 are compensable tasks, then $\Gamma_1;\Gamma_2$ denotes another compensable task while $\Gamma_{c_1} \bullet \Gamma_{c_2}$ denotes a task consisting of two distinct compensable subtasks. The control flow operators $\wedge, \vee$, and $\times$ as well as the t-calculus operators $||, \sqcap, \rightsquigarrow$ and $\otimes$ are associative.

Definition 4. *A time constrained Compensable Workflow net (CWF-net) N_c is a 5-tuple* (i, o, $\mathbb{T}$, $\mathbb{T}_c$, *F) such that:*

- *i is the input condition and* o *is the output condition,*
- *$\mathbb{T}$ is a set consisting of atomic tasks, and split and join tasks (i.e., routing tasks),*
- *$\mathbb{T}_c \subseteq \mathbb{T}$ is a set consists of the compensable tasks, and $\mathbb{T} \backslash \mathbb{T}_c$ is the set of uncompensable tasks,*
- *$F \subseteq (\{\boldsymbol{i}\} \times \mathbb{T}) \cup (\mathbb{T} \times \mathbb{T}) \cup (\mathbb{T} \times \{\boldsymbol{o}\})$ is the flow relation (for the net).*

The first compensable subtask of a compensable task is called the initial subtask; the compensation flow from the initial subtask is directed to an uncompensable task or the output condition followed by a compensable task, and every task in a workflow is on a directed path from `i` to `o`. The "Pain Assessment" workflow in the Case Study is a CWF-net. The underlying Petri net of a CWF-net (with time constraints) is a discrete time Petri net, denoted by TM, and defined formally as follows:

Definition 5. *An underlying time Petri net TM of a time constrained CWF-net $N_c = (\mathtt{i}, \mathtt{o}, \mathbb{T}, \mathbb{T}_c, F_N)$ is a 6-tuple (P, T, F, M_0, m^c, m^t) where,*

- *$P = P_A \cup P_D$, where P_A and P_D are finite sets of places and $P_A \cap P_D = \phi$,*
- *$T = \{t_1, t_2, ..., t_n\}$ is a finite set of transitions, each of which is associated with a set of pre-conditions $\{\psi^p\}$, and a set of actions, $\{\psi^a\}$, $P \cap T = \phi$ and $P \cup T \neq \phi$,*
- *$F = F_C \cup F_D$, where $F_C \subseteq (P_A \times T) \cup (T \times P_A)$ is the control flow relation, and $F_D \subseteq (P_D \times T) \cup (T \times P_D)$ is the data flow relation,*
- *M_0: $P \rightarrow \mathbb{N}$ is the initial marking,*
- *m^c is a function from transitions to time constraints; $m^c : T \rightarrow D_1 \times D_2$, where D_1, D_2 are sets of positive integers representing delays and durations, respectively,*
- *m^t: $T \rightarrow \mathbb{T}$ is a function, which maps a transition to a task.*

A marking of a Petri net is a function $M : P \rightarrow \mathbb{N}$. We say the marking assigns to each place a number of tokens.

Remark: The underlying time Petri net TM has one global clock to simulate the absolute time, *Now*; the *delay* and *duration* of each transition is simulated by an implicit clock, local to the transition. P_A is a set of activation places corresponding to places in the Petri net representation of the tasks of N_c and P_D is a set of data places (for data flow, F_D). Data places correspond to variables in the pre-conditions and actions of a time Petri net. $^\bullet t$ ($t^\bullet$) represents the set of input (output) places of t. Fig. 2 shows a small workflow fragment containing an XOR split, two atomic tasks and an XOR join task at the top, and its time Petri net representation at the bottom. For the operator $\times$, either $\tau_i\{\psi^a_{\tau_i}\}$ or $\tau_n\{\psi^a_{\tau_n}\}$ will execute.

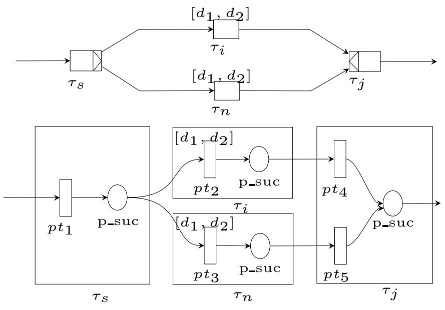


Fig. 2. Petri net representations of $\times$ (XOR)

Rule 1. *The firing rules of a time Petri net are as follows:*

1. *A transition t is said to be* **`enabled`** *if each input place p of t is marked with at least one token; formally, t is* **`enabled`** *iff, $\forall_{p \in {}^\bullet t}\ M(p) \geq 1$.*
2. *An* **`enabled`** *transition may or may not fire (depending on the firing constraints (delay and duration) of the transition).*
3. *A transition t (with $d_1 \in D_1$ and $d_2 \in D_2$) enabled by marking M at absolute time Now_{enable} cannot fire before absolute time ($Now_{enable} + d_1$) and can fire any time non-deterministically between ($Now_{enable} + d_1$) and ($Now_{enable} + d_1 + d_2$), unless disabled by the firing of some other transition.*
4. *A transition is* **`ready`***, if it can fire. If transition t_i fires, it leads the system to another state, at any time between ($Now_{enable} + d_1$) and ($Now_{enable} + d_1 + d_2$) (inclusive). A* **`ready`** *strong transition T_s must fire after ($Now_{enable} + d_1 + d_2$) is elapsed, and a* **`ready`** *weak transition will become disabled after that time. A firing of a* **`ready`** *transition t removes 1 token from each input place p of t, adds 1 token to each output place p of t, and resets the local virtual clock.*

$T_{ready(M)}$ denotes the set of `ready` transitions for a marking M. If a transition t is `ready` with marking M, $ready(t, M)$ is true, otherwise it is false.

3 Workflow Monitor Net

The following definition extends the notion of Petri net to account for monitoring CWF-nets.

Definition 6. *A Workflow Monitor Net (WMon-net) N_m associated to a CWF-net N_c is a 7-tuple (P, T, m^g, m^y, F, M_0, m^w) where, P and M_0 are same as in Definition 5 and:*

- $T = T^g \cup T^y \cup T^w$, where T^g, T^y and T^w are finite sets of transitions, $P \cap T = \phi$,
- $T^g = \{t_1^g, t_2^g, ..., t_n^g\}$ is a finite set of green transitions,
- m^g: $T^g \to \mathbb{T}$ is a function, mapping a green transition to a task of N_c,
- $T^y = \{t_1^y, t_2^y, ..., t_o^y\}$ is a finite set of yellow transitions,
- m^y: $T^y \to \mathbb{T}_c$ is a function, mapping a yellow transition to a compensable task of N_c,
- $T^w = \{t_1^w, t_2^w, ..., t_q^w\}$ is a finite set of white transitions, each of which is associated with a set of pre-conditions $\{\psi^p\}$ and a set of actions $\{\psi^a\}$,
- m^w is a function from white transitions to time constraints; $m^w : T^w \to D_1 \times D_2$ (D_1, D_2 are sets of positive integers representing delays and durations respectively),
- $F = F_C \cup F_D$, where $F_C \subseteq (P_A \times T^w) \cup (T \times P_A)$ is the control flow relation, and $F_D \subseteq (P_D \times T^w) \cup (T^w \times P_D)$ is the data flow relation.
A marking of a WMon-net is a mapping $M : P \to \mathbb{N}$.

Remark: The WMon-net uses the same global clock as the underlying TM of N_c to simulate the absolute time, and each white transition has its own local (implicit) clock.

Fig. 3 shows the graphical notation for workflow monitor elements. Note that from our definition of control flow, F_C, of a WMon-net, green and yellow transitions do not have any incoming arcs. They can only be connected with a place (activation place) by outgoing arcs by the definition of control flow. On the other hand, white transitions may be connected with places by both incoming and outgoing arcs.

A green transition is associated with the forward transition (pt_1 in Fig. 1) of an atomic (uncompensable or compensable) task in a CWF-net, and a yellow transition is associated with the compensation transitions (pt_2, pt_3 in Fig. 1) of an atomic compensable task in a CWF-net. Green and yellow transitions are virtual transitions and do not execute like ordinary transitions in a Petri net. We integrate a WMon-net with a CWF-net using these virtual transitions. The output places of green transitions and yellow transitions get tokens when the corresponding transitions (pt_1, pt_2, pt_3) execute in a CWF-net (see Fig. 4). Data places (P_D), and data flow relations (F_D) are not graphically represented in WMon-net.

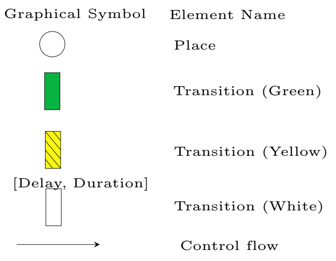


Fig. 3. Graphical representation of monitor components

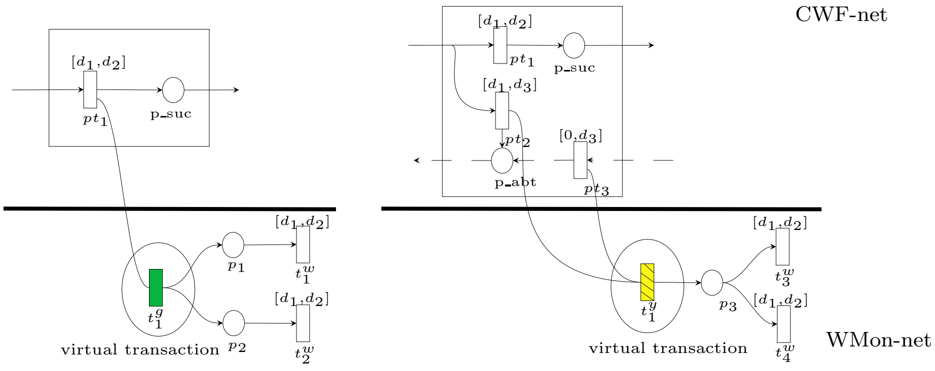


Fig. 4. A virtual transition

Definition 7. *A white transition t_i^w is said to be* **enabled** *if each input place p of t_i^w is marked with at least 1 token: i.e., t_i^w is* **enabled** *iff, $\forall_{p \in {}^\bullet t_i^w}\ M(p) \geq 1$, where ${}^\bullet t_i^w = \{p \mid p\ F_C\ t_i^w\}$ is the preset of t_i^w.*

t_i^w is said to be **ready** *if t_i^w exceeds its delay time and the pre-conditions are true: Formally, t_i^w is* **ready** *iff, $Now - enableTime(t_i^w) \geq delay(t_i^w)$, and $\forall_{\psi^p \in \psi_i^p} \psi_p = true$, where Now is the current time and $enableTime$ function returns the time when t_i^w was enabled,*

Rule 2. *The firing rules of a white transition in a WMon-net are as follows:*

1. *A* `ready` *transition may or may not fire (depending on whether or not the event actually takes place).*
2. *A firing of a* `ready` *transition t_i^w removes 1 token from each input place of t_i^w, adds 1 token to each output place of t_i^w and executes the actions of $\{\psi_i^a\}$.*

Virtual transitions (i.e., green and yellow transitions of WMon-nets) do not fire. The firing rules for the transitions of the TM underlying a CWF-net are as above, and they produce tokens in the output places of their corresponding virtual transitions in WMon-net. The variables in a pre-condition are shared variables, i.e., they are used by both the CWF-net and WMon-net. Because of shared variables in a pre-condition of a white transition, any changes made in a CWF-net are instantly visible to a WMon-net. An example is in the case study.

Now we give the definition of the `Tick` process (global clock) for the time Petri net underlying a CWF-net. It is important to note that the CWF-net and the WMon-net share the same `Tick` process.

Definition 8. *A* `Tick` *process is defined as follows:*

- *A* `Tick` *process is disabled iff there exists a transition t in the underlying TM of a CWF-net or WMon-net which is ready and exceeds its duration time: Formally, a* `Tick` *process is disabled iff, $\exists t \mid (ready(t) = true$ and $Now - enabled(t) \geq duration(t))$;*
- *A* `Tick` *process is enabled if it is not disabled;*
- *A* `Tick` *process may or may not increase the value of Now (current time) by 1 (depending on whether or not the time increased).*

3.1 Verification

In order to verify properties of a monitor (modeled as a WMon-net), using a model checker, the state space of the monitor must be combined with the state space of the workflow. Because a monitor monitors the processes of a workflow, it cannot generate its own state space without the input from a workflow. Therefore the two systems (i.e., workflow and monitor) must be combined and modeled in a model checker to generate the state space.

In [16], we provided an algorithm to translate a graphical model constructed using a workflow defined using CWML to DVE, the input language of the DiVinE model checker, greatly reducing the overall effort for model checking. Here we extend the algorithm to time constrained workflows (CWF-nets) with monitors, (WMon-nets). We begin with the definition of a DVE time Petri net and give details of the translation of the time Petri net (underlying the CWF-net) together with the WMon-net to the DVE time Petri net.

Definition 9. *A DVE time Petri net (DVE_T model) is a 7-tuple, $(now, Proc_{tick}, Proc, PTimer, V, T, S_0)$ where:*

- *"now" is a variable which indicates the current time,*
- *$Proc_{tick}$ is a clock process,*

- *$V \subseteq \{now\} \cup PTimer \cup \{v_1, v_2, .., v_k\}$ is a finite set of variables,*
- *$Proc = \{Proc_{tick}\} \cup \{Proc_1, Proc_2, ..., Proc_n\}$ is a finite set of processes,*
- *$PTimer = \{PTimer_1, PTimer_2, ..., PTimer_n\}$ is a finite set of variables, where $PTimer_i$ is the timer of $Proc_i$,*
- *$T = \{t_{tick}, t_1, t_2, ..., t_n\}$ is a finite set of transitions, where each transition is associated with a set of guards, G, and a set of effects, E,*
- *$S_0 : V \to \mathbb{N}$ (we assume $\mathbb{N}$ includes 0) is the initial variables assignment.*

Definition 10. *A state S of a DVT_T model is a function $S : V \to \mathbb{N}$.*

A state S in a DVE_T model transits to state S' if there is any change in $S(v_i)$.

Translation. The time Petri net TM = $(P_N, T_N, F_N, M_{N_0}, \mathrm{m}^c, \mathrm{m}^t)$ underlying a CWF-net N_c and an associated WMon-net $N_m = (P_M, T_M, \mathrm{m}^g, \mathrm{m}^y, F_M, M_{M_0}, \mathrm{m}^w)$ are translated to a DVE_T model $DM = (now, Proc_{tick}, Proc, PTimer, V, T, S_0)$ by the following rules:

- The global clock Now of a time Petri net represents the variable now;
- the tick process $Proc_{tick}$ works as defined in Definition 8;
- for each place $p \in \{P_N \cup P_M\}$, there corresponds a variable v_p in DM, the initial values of the variables are assigned with the initial marking M_{N_0} of TM and M_{M_0} of N_m. Formally, $\forall_{p \in P_N} S_0(v_p) = M_{N_0}(p)$ and $\forall_{p \in P_M} S_0(v_p) = M_{M_0}(p)$ and $V = \{now\} \cup \{v_i \mid 1 \leq i \leq \mid P_N \mid + \mid P_M \mid\}$;
- $PTimer$ is a set of local variables representing the implicit local clocks in T_N and T_M. The values of the variables hold the enabling time of tasks. So the local clocks are derived from the difference from the *now* variable. For this reason, there is no drift from the global clock;
- for each transition $t_i \in T_N$, which is associated with a set of pre-conditions $\{\psi_i^p\}$ and a set of actions $\{\psi_i^a\}$, there corresponds a process $Proc_i$ in DM; $Proc_i$ has a transition $t_i' \in T$ associated with a set of guards, G_i and a set of effects, E_i; the *guards* and *effects* of t_i' are determined by the pre-conditions, actions, time constraints and flow relations of t_i and virtual transitions of N_m (i.e., if t_i is virtually connected to a green or yellow transition):
  $G_i = \{\psi_i^p\} \cup \{S(v_p) \geq 1 \mid p \in {}^\bullet t_i\} \cup \{(now - PTimer_i) \geq d_1(t_i)\}$;
  $E_i = \{\psi_i^a\} \cup \{S(v_p) = 1 \mid p \in t_i^\bullet\} \cup \{PTimer_k = now | (t_i^\bullet \cap {}^\bullet t_k) \neq \phi$, where $t_k \in T_N\} \cup \{S(v_p) = 1 \mid p \in t^{g\bullet}$, where $\mathrm{m}^g(t^g) = \mathrm{m}^t(t_i)\ \} \cup \{S(v_p) = 1 \mid p \in t^{y\bullet}$, where $\mathrm{m}^y(t^y) = \mathrm{m}^t(t_i)\ \}$;
- for each transition $t_i^w \in T^w \subset T_M$, there corresponds a process $Proc_i$ in D; $Proc_i$ has a transition t_i' associated with a set of guards, G_i and a set of effects, E_i; the *guards* and *effects* of t_i' are determined by the pre-conditions, actions, time constraints and flow relations of t_i^w:
  $G_i = \{\psi_i^p\} \cup \{S(v_p) \geq 1 \mid p \in {}^\bullet t_i^w\} \cup \{(now - PTimer_i) \geq d_1(t_i^w)\}$;
  $E_i = \{\psi_i^a\} \cup \{S(v_p) = 1 \mid p \in t_i^{w\bullet}\} \cup \{PTimer_k = now | (t_i^{w\bullet} \cap^\bullet t_k^w) \neq \phi$, where $t_k^w \in T^w\}$;
- a transition t_i' in a DVE_T model is **ready** if it satisfies its *guard* conditions;
- if t_i' is ready at state S, $ready(t_i', S)$ is true, otherwise it is false; A **ready** transition may or may not fire depending on whether or not the event actually takes place).

Proof of Correctness. The yellow and green transitions of a WMon-net are virtual transitions used to make the association of the time Petri net underlying a CWF-net. Since a WMon-net is also a time Petri net, a time Petri net represents the combination of a CWF-net with a WMon-net; our discussion below therefore simply considers a time Petri net and its translation to a DVE_T model.

Definition 11. *A state in a time Petri net TM is a tuple, $S^E = \langle M, Now\rangle$ where: M denotes the current marking and Now represents the absolute time. The initial state S_0^E consists of the initial marking M_0 and absolute time zero. A state S_i^E can transit to state $S_{i'}^E$, by either (1) changing the marking by firing a transition, or (2) increasing the value of global clock value.*

Let $\pi = S_0^E, S_1^E ...$ be a path in a time Petri net, let ϕ be LTL formulae, and p be a propositional variable. The notation $TM, \pi \vDash \phi$ means that the formula ϕ holds or is satisfied along the path π in the model TM. We say TM satisfies the formula ϕ, denoted as $TM \vDash \phi$, iff all of its runs, emanating from the initial state S_0^E, satisfy ϕ. The satisfaction relation, $\vDash$ for a DVE_T model DM is identical to the satisfaction relation of a time Petri net (see [13]).

Definition 12. *A state S_i^E of a time Petri net TM, and a state S_i of a DVE$_T$ model DM are equivalent, denoted $S_i^E \cong S_i$ iff:*

1. $\forall_{p\in P}, M_i(p) = S_i(var_p)$, where var_p is the variable corresponding to place p;
2. The value of TM's global clock Now equals $S_i(now)$.

Definition 13. *A path $\pi = S_0^E, S_1^E ...$ in a time Petri net TM and a path $\pi' = S_0 S_1 ...$ in a DVE$_T$ model DM correspond (written as $\pi \cong \pi'$) iff $\forall i \geq 1$ $S_i^E \cong S_i$.*

Definition 14. *A time Petri net $TM = (P, T, F, M_0, m^c, m^t)$ and a DVE$_T$ model $DM = (now, Proc_{tick}, Proc, PTimer, V, T, S_0)$ are equivalent ($TM \cong DM$) iff: 1. $S_0^E \cong S_0$, 2. for every path starting from S_0^E ($\pi = S_0^E S_1^E ...$) in TM there is a corresponding path in DM starting from S_0, ($\pi' = S_0 S_1 ...$) and for every path starting from S_0 in DM there is a corresponding path in TM starting from S_0^E.*

Theorem 1. *If DM is the DVE_T translation of a time Petri net TM, then $TM \cong DM$.*

Proof: Let TM be a time Petri net and DM be its DVE$_T$ model; We show $TM \cong DM$. Let π be a path in TM; we will show by induction on the length of the path (=number of states in π) that π', the translation of π, corresponds to π.

Base Case: Show: $S_0^E \cong S_0$.
The initial state S_0^E of TM and S_0 of DM are equivalent as: $\forall_{p\in P} M_0(p) = S_0(var_p)$ and the value of TM's global clock Now equals $S_i(now)$, as the value of Now at state S_0^E is 0 and $S_0(now) = 0$; hence $S_0^E \cong S_0$.

Induction Step: Show that if for all paths π of length k in TM there is a corresponding path π' of length k in DM, then for all path of length $k+1$ in TM, there is a corresponding path π' of length $k+1$ in DM.

Let us assume that π is a path of length k in TM and π' is a corresponding path of length k in DM. Thus $S_k^E \cong S_k$ (induction hypothesis). So $\forall_{p\in P} M_k(p) = S_k(var_p)$, and $S_k^E(Now) = S_k(now)$. According to Definition 12, a state can transit to another state either 1. by firing a transition to change the marking or 2. by increasing the value of the global clock.

1. For each ready transition in TM, there is a ready process in DM. Let M_k be the marking at state S_k^E. Since the *guard* condition of a transition in DM is the same as the *ready* condition of a transition in a time Petri net, we have: $\forall_{t\in T_{ready(M_k)}} ready(Proc_t, S_k) = true$. If any transition $t \in T_{ready(M_k)}$ fires, we get the following changes to the marking: $\forall_{p\in {}^\bullet t} M_{k+1}(p) = M_k(p) - 1$, and $\forall_{p\in t^\bullet} M_{k+1}(p) = M_k(p) + 1$. In the DVE$_T$ process $Proc_t$ for the transition t, the transition t' will change the values of the variables as follows: $\forall_{var_p\in {}^\bullet t'} S_{k+1}(var_p) = S_k(var_p) - 1$, and $\forall_{var_p\in t'^\bullet} S_{k+1}(var_p) = S_k(var_p) + 1$. By the Translation, we conclude: $\forall_{p\in P}\ M_{k+1}(p) = S_{k+1}(var_p)$. Since the clock remains unchanged, $S_{k+1}^E(Now) = S_k^E(Now) = S_k(now) = S_{k+1}(now)$.

2. If the global clock of TM has not been disabled then according to Definition 8, the state S_k^E transit to the state S_{k+1}^E by increasing Now by 1, so $S_{k+1}^E(Now) = S_k^E(Now + 1)$. A disabled global clock can not increment the clock. The DVE$_T$ process $Proc_{tick}$ increases the global time denoted by variable now. The disabling condition of Process $Proc_{tick}$ is same as in Definition 8 and a disabled $Proc_{tick}$ process can not increment the now variable. Thus, $S_{k+1}(now) = S_k(now + 1)$ By the Translation, we conclude: $S_{k+1}^E(Now) = S_{k+1}(now)$. Since the global clock is increased, there is no change in the marking. So, $\forall_{p\in P} M_k(p) = \forall_{p\in P} M_{k+1}(p) = S_k(v_p) = S_{k+1}(v_p)$

From 1 and 2 we conclude $S_{k+1}^E \cong S_{k+1}$; so there is a path π' of length $k+1$ in DM corresponding to the path π of length $k+1$ in TM. Thus for every path $\pi \in TM$ starting from S_0^E, there is a corresponding path $\pi' \in DM$ starting from S_0. Similarly we can show that for every path π' in DM starting from S_0, there is a corresponding path π in TM starting from S_0^E. We conclude: $TM \cong DM$.

The proof of the following proposition is a straightforward structural induction on the length of the formula ϕ, recalling that if two paths π and π' correspond then for all i, the suffixes π^i and π'^i correspond.

Proposition 1. *Let π be a path in a time Petri net TM corresponding to a path π' in DM. Then for any LTL formula ϕ, $\pi \vDash \phi$ iff $\pi' \vDash \phi$.*

Theorem 2. *(Correctness) Let ϕ be an LTL formula, and let TM be a time Petri net and let DM be its DVE_T tranlation. Then $TM \vDash \phi$ iff $DM \vDash \phi$.*

Proof: Suppose $TM \vDash \phi$ for some LTL formula ϕ. Then for all paths π in TM starting at S_0^E, $\pi \vDash \phi$. We show that for all paths π' in DM starting at S_0, $\pi' \vDash \phi$. Let π' be a path in DM starting at S_0. Since $TM \cong DM$, there is a path π in TM starting at S_0^E and corresponding to π'. By assumption, $\pi \vDash \phi$.

The previous proposition allows us to conclude that $\pi' \vDash \phi$. Analogously we can show that if $DM \vDash \phi$ then $TM \vDash \phi$.

4 Case Study

We model a workflow and a monitor system following the guidelines for the management of cancer related pain of adults [7]. The guideline starts with the screening of all cancer patients for pain. If 'pain' is identified as a focus of care, a **Pain Assessment and Care Plan** is conducted. All causes of pain, pain intensity, pain location, and other symptoms are assessed; full history of patient's physical exams and previous analgesics and response to each are documented. Patient's Opioid Regimen (i.e., Non-Opioid, Weak Opioid, Strong Opioid Regimen) is determined based on patient's 'Pain intensity', 'Stability', and 'Current Medication'. If a patient is under a Strong Opioid, a pain reassessment should be done after a certain interval. For the **Strong Opioid Regimen**, the guideline suggests continuing the current analgesic and reassessing within a week if a patient is responding (i.e., current pain level is less than previous pain level); it suggests a different reassessment interval depending on the current pain level if a patient is not responding. The guideline suggests increasing dose by 10% q4h ATC (every 4 hour around the clock) and reassessing at least every 48–72 hours for **Mild Pain**; increase dose by 10–25% q4h ATC and reassess at least every 24 hours for **Moderate Pain**; increase dose by 25–50% q4h ATC and reassess at least every 12 hours for **Severe Pain**. 'Opioid toxicity' or 'side effects' may be managed by *Opioid Rotation*, *Addition of Adjuvant(s)*, *Opioid Dose Reduction*, or *Change Route of Opioid*. 'Opioid toxicity' refers to symptoms related to opioid dose (e.g., neurological symptoms) whereas 'side effects' are symptoms which may occur at any opioid dose (e.g., constipation, nausea and vomiting, sedation, myoclonus hyperalgesia/alodynia, hallucinations, delirium, nightmares).

Fig. 5 shows the "Pain Assessment" workflow (a time constrained CWF-net), the "Strong Opioid Regimen" subnet workflow (a time constrained compensable CWF-net), the "Reassessment" monitor (a WMon-net), and the "Side effect management" monitor (a WMon-net) we modeled. A key defines those elements from CWML$_T$ used in these models (motivated by a common workflow modeling language such as YAWL). ASSESS_PAIN task assesses all causes of pain, determines pain location(s), pain intensity, and other symptoms. Historical information is gathered in the COLLECT_HISTORY task. All previous analgesics (including opioids) and response to each are documented in the RECORD_PREVIOUS_ANALGESICS task. Depending on the patient's pain intensity, stability and current medication, an opioid regimen is assigned. While the patient is treated under one of the opioid regimens, a regular assessment is done after a certain interval. The care delivery and reassessment processes are surrounded by a loop which continues as long as the patient is involved in this healthcare protocol after which the patient is DISCHARGED from the protocol; the "Reassessment" monitor observes the

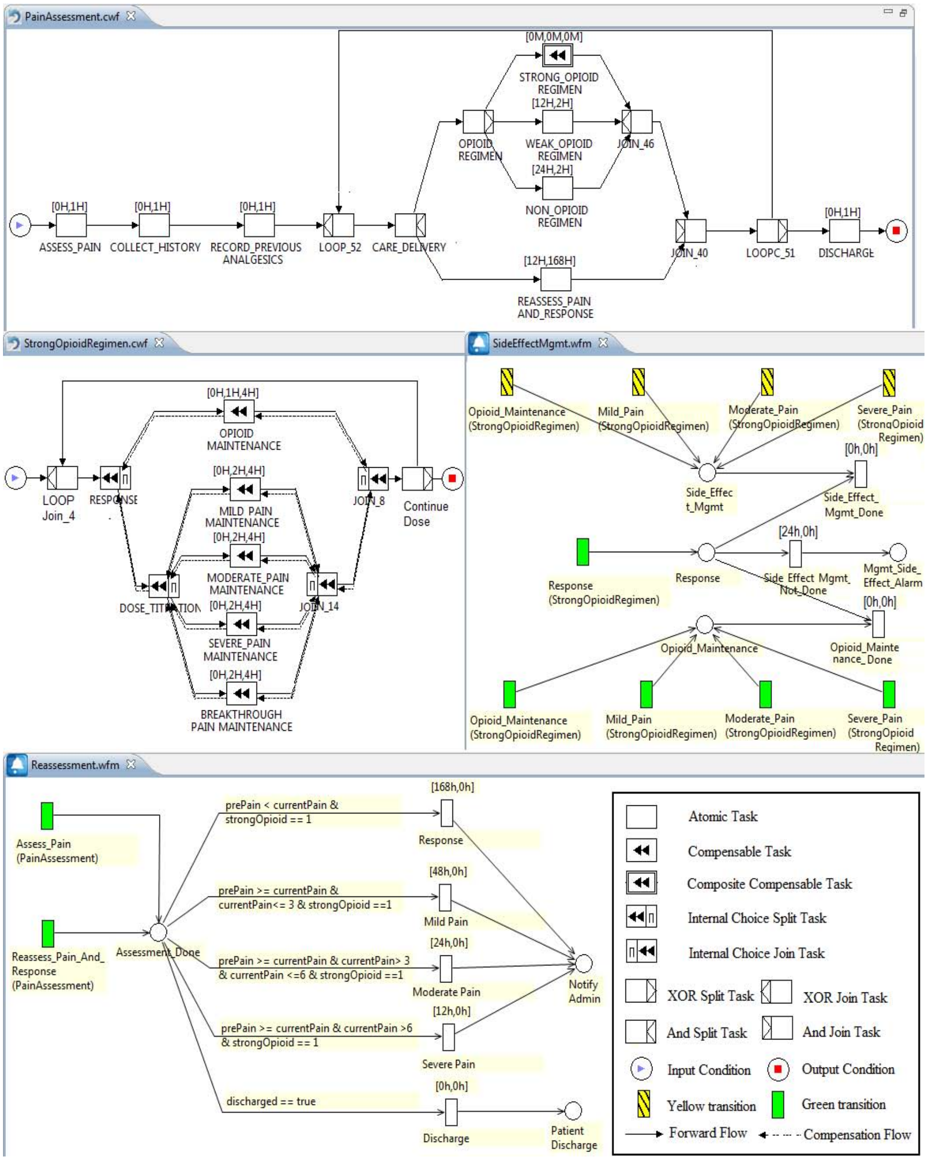


Fig. 5. "Pain Assessment workflow", "Strong Opioid Regimen subnet workflow", "Side effect management monitor", and "Reassessment monitor"

interval of assessment and notifies the clinician if a reassessment is not done within the time suggested by the guideline. This monitor contains two green transitions: Assess_Pain (connected with the ASSESS_PAIN task of the "Pain Assessment" workflow) and 'Reassess_Pain_And_Response' (connected with the REASSESS_PAIN_AND_RESPONSE task). When the task ASSESS_PAIN or

REASSESS_PAIN_AND_RESPONSE executes, it generates a token in place 'Assessment_Done' in the "Reassessment monitor". The white transitions connected with the place 'Assessment_Done' become enabled and wait until the delay time has passed. If another reassessment is done before any of the white transitions fires, the token in place 'Assessment_Done' is reset and the white transitions start counting again. But if another reassessment is not done within due time, an administrator will be notified. Note that the pre-conditions of the transitions use shared variables (e.g., prePain, currentPain, strongOpioid, etc.); the values of these variables are set from the "Pain Assessment" workflow.

The "Strong Opioid Regimen" subnet workflow has been modeled with compensable components. While a patient is administered a dose, there might be Opioid toxicity or other side effects. The guideline suggests some opioid toxicity or side effect management options; during the execution of any of the tasks *Opioid*, *Mild Pain*, *Moderate Pain*, *Severe Pain*, *Breakthrough Pain Maintenance*, if the clinician observes any of the side effects or opioid toxicity, the workflow allows her to perform the appropriate option to comfort the patient as compensation. The execution of the options are monitored by the "Side effect management" monitor. In the modeling, the dotted arcs represent the backward flow. Whenever a compensable task is aborted (i.e., cannot finish the forward flow or normal business logic), it compensates and activates the backward flow. Using compensation as a fault tolerance mechanism reduces the size of the graphical model. The backward flow reaches the previous compensable task and activates its compensation. This continues until the compensation flow reaches the initial compensable task, from where the forward flow continues.

The "Side effect management" monitor contains 4 yellow and 4 green transitions connected with the *Opioid*, *Mild Pain*, *Moderate Pain*, *Severe Pain*, *Breakthrough Pain Maintenance* compensable tasks. The monitor also contains a green transition connected with the forward flow of the RESPONSE task of the "Strong Opioid Regimen" subnet. When the RESPONSE task executes, it generates a token in place 'Response' (in the monitor). If any of the tasks from *Opioid*, *Mild Pain*, *Moderate Pain*, *Severe Pain*, *Breakthrough Pain Maintenance* either executes (associated green transition fires) or compensates (associated yellow transition fires), the token in the place 'Response' is consumed by the appropriate transition 'Side_Effect_Mgmt_Done' or 'Opioid_Mgmt_Done'. But if the token remains in the place for more than 24 hours, it will be consumed by the transition 'Side_Effect_Mgmt_Not_Done' and an alert will be sent. Some of the properties we model checked are provided below with their LTL formulas:

- Prop1: If a patient is under Strong Opioid Regimen, not responding (to medication), current pain level is > 6 and another reassessment is not done within 12 hours, the clinician gets a notification (G ((strong_opioid_patient && patient_is_not_responsive && cur_p_level_gr_than_six && reassessed_twelve_hrs_ago) − > F (clinician_is_notified))).
- Prop2: If a patient is discharged, the clinician will not receive any reassessment alert (G (patient_discharged − > ! F (clinician_is_notified))).

– Prop3: If a patient is not under Strong Opioid Regimen, current pain intensity is in between 2–3 but the pain is not stable then the patient is moved to Weak Opioid Regimen (G (not_strong_opioid_patient && cur_p_level_btn_2_a- nd_3 && pain_is_not_stable − > weak_opioid_patient))
– Prop4: If a patient is on Weak Opioid with moderate-severe pain, the patient is moved to Strong Opioid Regimen (G (weak_opioid_patient && cur_p_level_btn_4_and_6 − > strong_opioid_patient))
– Prop5: If required 'Side effects' are not managed within 24 hours, the clinician gets a notification (G (response_measured && side_effect_not_managed && resp_measured_24_hrs_ago − > mgmt_side_effect_alarm)).

The results are shown in Table I; The first time we ran the model checker, the results showed that the first two properties were false, and provided counter examples (this is denoted by ACC Cycle). For the first property the counter example showed that even if the conditions were true then the clinician may not get a notification when the patient was discharged. For the second property, the counter example showed that the clinician receives the notification if the discharge operation executes at the same time as a notification was supposed to be sent. It was clear from the counter examples that there was a flaw in the model, which was that the patient's discharge was not taken into consideration in the pre-conditions of *Response*, *Mild Pain*, *Moderate Pain* and *Severe Pain* transitions. As a result while the *Discharge* transition is ready, other transitions could possibly be ready and execute. The initial model was corrected by rewriting the pre-conditions and subsequent model checking showed that both properties were satisfied.

Table 1. Verification Results

Property	*ACC Cycle*	*State*	*Memory (MB)*	*Time (s)*
Prop1	Yes	3172	86.1	1.3
Prop2	Yes	7169	134.4	2.1
Prop3	No	23781703	2693.2	16.5
Prop4	No	20398348	2147.0	14.0
Prop5	No	25934140	2974.3	17.7

Since the general knowledge base for medicine is very large, and frequently organized as an ontology, our system is integrated with an ontology to illustrate its use. We designed a small ontology in OWL 2.0 representing the facts and rules about strong opioids needed for determining reassessment time. We integrated the ontology with the monitor system and used the FaCT++ reasoner. During execution, the 'Opioid Regimen' xor-split task generates a query with the list of medication that a patient is taking, and the current and previous pain levels as parameters and sends them to the FaCT++ reasoner. The reasoner computes whether the medicine is a strong opioid and returns the suggested reassessment time. (Space does not permit us to include full details).

5 Related Work

During the last decade various work has been done on workflow monitoring. In [14] the authors presented a research project *Palliasys* for continuously monitoring the health status of palliative patients and sending alerts to doctors when some conditions of the state of a patient are met. This approach is not workflow based (rather agent based) and the query language for defining a monitor is not graphical and is weak for writing temporal constraints (i.e., only 'days' can be specified); the formal verification of the system was missing, and no ontology was integrated.

In [15] the authors developed a process analysis tool (PISA) that can be employed to analyze the audit trail data of different workflow management systems in conjunction with target data from business process modeling tools. A prototype implementation was made that integrates data of the ARIS Toolset and IBM MQSeries Workflow. The authors focused on the continuous process engineering cycle, particularly on *Process Change Management* as opposed to runtime monitoring to generate real time alerts of a process involving patients and clinicians in a highly variable environment.

In [2], the authors presented BMon (Business process Monitoring), a query language and system for monitoring business processes. BMon allows users to visually define monitoring tasks and associated reports, using a simple intuitive interface similar to those used for designing BPEL processes. It is not possible to specify temporal constraints in BMon and a formal verification for time constrained workflow with a monitor designed in BMon is not possible.

Simmonds et al. presented the tool RuMoR in [17] which performs monitoring of web service applications, and, when violations are discovered, automatically proposes and ranks recovery plans. Properties, specified using property patterns, are transformed into finite state automata. While runtime monitoring, some compensation mechanism, and verification are common to our system there are differences. RuMoR takes a BPEL program as input and translates to a labeled transition model using WS-Engineer. Monitors are specified as finite-state automata. Although data aware verification can be done in RuMoR it has limitations with respect to time. RuMoR was implemented within the IBM WebSphere using the interception mechanism, whereas the architecture of NOVA Workflow is light-weight as it uses Spring and aspect oriented programming techniques enabling its use with various J2EE application servers [12]. In addition, NOVA Workflow uses an ontology for decision support.

There are various tools for the analysis of workflow models using Petri net based semantics. For example *Reo* is a graphical channel-based coordination language that enables the modeling of complex behavioral protocols using a small set of channel types with well-defined behavior [9]; TINA [5] and Romeo [8] are frameworks for the verification of time Petri nets. It is possible to design a timed compensable workflow and monitor system with these Petri net based tools or by synchronizing the execution of certain pairs of transitions (e.g., using a channel based approach) but we have developed a high level graphical language for time compensable workflow with monitors which significantly reduces the difficulty of designing a large workflow and monitor system. Moreover, the data aware verification method we presented here using the parallel distributed model checker DiVinE provides excellent support to verify large systems. We note that while UPPAAL [3] is a popular timed automata model checker, the distributed version of UPPAAL is under development.

In [6], the author pointed out some limitations of Petri nets including: 1) Petri nets may lose track of the relations between resources and their functionalities.

2) Petri nets may be difficult to relate the dynamic behavior to the static relation between cases. 3) Petri nets may not be a good choice for modeling collaborative workflows that are typical for integrated health informatics for effective sharing of information. 4) Mechanisms of exception handling are complicated. Our use of a high level workflow language (i.e., CWML_T, WMon-net) overcomes many of the limitations (e.g., resource planning is handled by the workflow engine) and the compensation mechanism which is easily modeled in CWML_T is capable of handling complicated exceptions.

6 Conclusion and Future Work

Healthcare processes involve many services and provide numerous settings for care. Monitoring the processes and notifying relevant caregivers for any delayed tasks or unexpected situations will help ensure the quality and continuum of care. To design, develop, and verify a monitor system for a time constrained workflow is difficult using existing tools. Using a graphical modeling language for modeling a monitor system and integrating it with a workflow system via virtual transitions can overcome many of these difficulties while giving modularity of design. A time constrained workflow with monitor can be easily modeled using a high level graphical modeling language and verified automatically using the methods described in this paper. Modeling, and verifying workflow systems with monitors will help developers build error free systems especially health care applications.

Scheduling is another important aspect of healthcare and other safety-critical systems. In our lab, we are doing research on scheduling and resource management of healthcare workflows. A reduction mechanism to reduce the state space to verify large (timed) workflows is also under development. An interdisciplinary team in the StFX Centre for Logic and Information[2], consisting of researchers and students in computer science and health related fields, have been working closely with clinicians, administrators, and other health care providers from the Guysborough Antigonish Strait Health Authority (GASHA) to greatly refine the details of the process of care and include information on time, access control and other process specific information.

Acknowledgment. This research is sponsored by NSERC of Canada and by the Atlantic Canada Opportunities Agency. Computational facilities are provided by ACEnet. We thank Prof. Heather Jewers from the StFX School of Nursing, and Madonna MacDonald (VP, Community Services) and clinicians from GASHA.

References

1. DiVinE project, `http://divine.fi.muni.cz/` (last accessed on November 2010)
2. Beeri, C., Eyal, A., Milo, T., Pilberg, A.: Query-based monitoring of BPEL business processes. In: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, SIGMOD 2007, pp. 1122–1124. ACM, New York (2007)

[2] `http://logic.stfx.ca/`

3. Bengtsson, J., Larsen, K.G., Larsson, F., Pettersson, P., Yi, W.: Uppaal — a Tool Suite for Automatic Verification of Real–Time Systems. In: Alur, R., Sontag, E.D., Henzinger, T.A. (eds.) HS 1995. LNCS, vol. 1066, pp. 232–243. Springer, Heidelberg (1996)
4. Berthomieu, B., Diaz, M.: Modeling and verification of time dependent systems using time Petri nets. IEEE Trans. Softw. Eng. 17, 259–273 (1991)
5. Berthomieu, B., Vernadat, F.: Time petri nets analysis with TINA. In: Proceedings of the 3rd International Conference on the Quantitative Evaluation of Systems, pp. 123–124. IEEE Computer Society, Washington, DC (2006)
6. Bertolini, C., Liu, Z., Schaf, M., Stolz, V.: Towards a formal integrated model of collaborative healthcare workflows. UNU-IIST Report No. 450 (June 2011)
7. Broadfield, L., Banerjee, S., Jewers, H., Pollett, A., Simpson, J.: Guidelines for the management of cancer-related pain in adults. Supportive Care Cancer Site Team, Cancer Care Nova Scotia (2005)
8. Lime, D., Roux, O.H., Seidner, C., Traonouez, L.-M.: Romeo: A Parametric Model-Checker for Petri Nets with Stopwatches. In: Kowalewski, S., Philippou, A. (eds.) TACAS 2009. LNCS, vol. 5505, pp. 54–57. Springer, Heidelberg (2009)
9. Kokash, N., Krause, C., de Vink, E.P.: Time and data-aware analysis of graphical service models in Reo. In: Fiadeiro, J.L., Gnesi, S., Maggiolo-Schettini, A. (eds.) SEFM, pp. 125–134. IEEE Computer Society (2010)
10. Lamport, L.: Real-Time Model Checking Is Really Simple. In: Borrione, D., Paul, W. (eds.) CHARME 2005. LNCS, vol. 3725, pp. 162–175. Springer, Heidelberg (2005)
11. Li, J., Zhu, H., Pu, G., He, J.: A formal model for compensable transactions. In: Proceedings of the 12th IEEE International Conference on Engineering Complex Computer Systems, pp. 64–73. IEEE Computer Society, Washington, DC (2007)
12. MacCaull, W., Rabbi, F.: NOVA Workflow: A workflow management tool targeting health services delivery. In: International Symposium on Foundations of Health Information Engineering and Systems, FHIES 2011, Johannesburg, South Africa, pp. 74–91 (2011)
13. Mashiyat, A.S., Rabbi, F., MacCaull, W.: Modeling and Verifying Timed Compensable Workflows and an Application to Health Care. In: Salaün, G., Schätz, B. (eds.) FMICS 2011. LNCS, vol. 6959, pp. 244–259. Springer, Heidelberg (2011)
14. Moreno, A., Valls, A., Riaño, D.: Palliasys: agent-based proactive monitoring of palliative patients. In: IWPAAMS (2005)
15. Muehlen, M.Z., Rosemann, M.: Workflow-based process monitoring and controlling - technical and organizational issues. In: Proceedings of the 33rd Hawaii International Conference on System Science (HICSS-33), pp. 1–10. IEEE Computer Society Press (2000)
16. Rabbi, F., Wang, H., MacCaull, W.: Compensable WorkFlow Nets. In: Dong, J.S., Zhu, H. (eds.) ICFEM 2010. LNCS, vol. 6447, pp. 122–137. Springer, Heidelberg (2010)
17. Simmonds, J., Ben-David, S., Chechik, M.: Guided recovery for web service applications. In: Proceedings of the ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE 2010, pp. 247–256. ACM, New York (2010)

Position Paper: Researching and Developing Open Architectures for National Health Information Systems in Developing African Countries

Deshendran Moodley[1,2,4], Anban W. Pillay[1,4], and Christopher J. Seebregts[1,2,3,4]

[1] School of Computer Science, University of KwaZulu-Natal, Durban, South Africa
[2] Jembi Health Systems, Cape Town and Durban, South Africa
[3] Medical Research Council, Cape Town, South Africa
[4] UKZN/CSIR Meraka Centre for Artificial Intelligence Research

Abstract. Most African countries have limited health information systems infrastructure. Some health information system components are implemented but often on an adhoc, piecemeal basis, by foreign software developers and designed to solve specific problems. Little attention is usually paid to how these components can fit into an integrated national health information system and interoperate with other components. The Health Enterprise Architecture Laboratory was recently established in the School of Computer Science at the University of KwaZulu-Natal in South Africa to undertake research and build capacity in open health architectures for developing African countries. Based on field experiences and requirements in South Africa, Mozambique and Rwanda, the laboratory is evolving a generic Health Enterprise Architecture Framework and Repository of Tools specifically for low resource settings. In this paper we describe these three initiatives and the expected impact on implementing health information systems in developing African countries.

Keywords: health enterprise architectures, ICT for developing countries, interoperability, ontologies, postgraduate training, open architectures.

1 Introduction

Health information systems (HIS) are a critical component of well-functioning health systems [27] and an important pre-requisite to improving health outcomes [1][11]. The worldwide growth in information and communication technology is opening up important opportunities to strengthen HIS and many efforts are underway to develop national health information systems (NHIS) [19][26]. Developing countries are also undergoing rapid technological transformation associated with the increased availability of mobile devices and network connectivity, among others, that are accelerating adoption of HIS [12]. While significant potential exists to leverage ICTs to improve equitable health service delivery and health outcomes in remote and underserved areas, many challenges remain.

Developing countries have unique environments in which limited availability of infrastructure, specialized technical skills and financial resources impact on the development of systems [18] [25]. In addition, the social, economic and political environments as well as the communication infrastructure are often prone to rapid change [9] [8]. Scarce resources are not optimally deployed to achieve potential cost savings and economies of scale. Limited resources are understandably directed to NHIS components with a single purpose that have a more immediate impact on health outcomes. Little attention is paid to issues such as support for open standards, interoperability and reusability across different systems. There is no well-developed notion of an open architectural framework to facilitate the design and rapid implementation of interoperable and sustainable NHIS [26]. Such a framework will be vendor-independent, based on open standards and allow various stakeholders to create add-on components or customizations and extensions that increases flexibility, functionality, interoperability and capabilities to suit individual requirements.

The application of an open architectural approach has significant potential to enable a paradigm shift in NHIS development. A systematic approach to high-level information system design at national level will assist African Ministries of Health to better utilize resources available for independent projects implemented by donors, non-governmental organizations and universities. Common paradigms or "architectural patterns" can be used as metaphors that allow the knowledge gained from one implementation to be used in others.

Several initiatives in the developed world are underway that focus on architectural approaches to building NHIS. However, given the unique environments of African countries, it cannot be assumed that technical solutions that have been successfully deployed in the developed world can be easily transferred to these environments without performing fundamental re-engineering or customization.

The Health Enterprise Architecture Laboratory (HEAL) was recently established in the School of Computer Science at the University of KwaZulu-Natal in South Africa to undertake research and build capacity in open architectures and technologies for NHIS in developing African countries. Based on field experiences and requirements in South Africa, Mozambique and Rwanda, HEAL is currently developing a Health Enterprise Architecture Framework (HEAF)[1] and Repository of Tools (HEART) specifically for low resource settings. The HEAF and the HEART will capture and distill best practices and experiences from these environments that can facilitate the development of NHIS in other low resource countries. The aim is to establish the HEAF and HEART as community-driven projects to be continually informed by user experiences and form the first tools in a generalized open architectural framework.

In this paper we describe these three initiatives and the potential impact in developing African countries. In section 2 we analyze the current landscape and

[1] https://sites.google.com/site/heafproject/

research challenges and in section 3 we describe the HEAL. In section 4 the HEAF is described and in section 5, the HEART. In Section 6 we summarize the potential impact of these initiatives and draw our conclusions.

2 National HIS in Africa: The Current Landscape and Future Prospects

African countries differ widely in terms of their current HIS infrastructure and NHIS strategies. Typical NHIS include health, pharmacy and laboratory subsystems. Applications cover a wide variety of health services, including preventative, curative, supply chain and financial services. An ISO expert group, in partnership with the World Health Organization, is developing a technical report to help developing countries implement and harmonize NHIS[2]. The report describes the characteristics of the core subsystems and provides a maturity model for the implementation of NHIS in low resource settings. Typically, countries begin with population-based information systems, providing data on national health issues and guiding policy decisions at a national level. Once established, they develop systems that are more focused at the person level.

Our experiences are based on three African countries that differ widely in terms of their level of maturity of NHIS implementation but yet have many common characteristics.

South Africa is one of the most well developed African countries and its NHIS reflects characteristics of both high and low resource settings. The larger cities have advanced HIS technologies while most provinces in the country have deep rural settings with little or no computerized information systems. Broadband network and mobile phone penetration is well developed but focused on the urban areas.

Rwanda is one of the smallest and most densely populated African countries. The country has a progressive and expansive eHealth policy. The national eHealth coordination unit is actively driving the deployment of a number of advanced systems to strengthen the NHIS. These include patient and hospital information systems, community health information systems using mobile phones, and a health information exchange implementation project.

Mozambique is still developing much of its HIS. Working in collaboration with a local organization (MOASIS[3]) attached to the University of Eduardo Mondlane, the Ministry of Health has implemented several systems, including an aggregated data system and computerized death registration system based on a limited list of codes from the International Classification of Diseases[4]. A national health enterprise architecture project is underway that is documenting a baseline architecture and developing a target architecture in line with the national eHealth strategic plan. Other projects under development include a national data warehouse project and patient-based systems.

2 http://www.iso.org/iso/pressrelease.htm?refid=Ref1275
3 http://www.moasis.org.mz
4 http://www.who.int/classifications/icd/en/

Based on experiences implementing HIS in these countries we have identified the following common requirements:

- *Urgent Deployment.* African countries are ravaged by a high burden of disease and have an urgent requirement to rapidly deploy applications to facilitate, manage and optimize health interventions that have an immediate impact on health outcomes in critical disease areas.
- *Balance between Innovation and Pragmatism.* It is difficult and risky to implement systems requiring substantial change to existing operations and workflows. Yet, a high level of innovation is required to introduce and leverage the latest advances in computer science and to take advantage of the rapidly improving ICT landscape in a way that makes optimal impact. Deployment of disruptive or ineffective systems can have disastrous consequences for an already overburdened health system.
- *Evidence-Based and Effective.* Many systems are developed from first principles with little regard to what has been done before. This may be due to limitations in documented evidence and evaluation. Wherever possible, systems should be based on evidence of the effectiveness of previous systems, while still leveraging the latest techniques and technologies.
- *Sustainable and Affordable.* It is essential that systems are sustainable, affordable and harmonized with a country's strategic plans. Good design and rigor must be balanced with rapid engineering, to meet immediate needs and ensure buy-in from non technical stakeholders while also giving due consideration to long term sustainability. Special consideration must be given to developing the specialized IT skills necessary to maintain affordable and cost-effective systems.
- *Support Practical Modes of Operation.* Systems must include both computerized and manual, paper-based solutions as well as network-connected and network-disconnected scenarios.

Development of an open architectural framework is a necessary first step to facilitate the design and rapid implementation of effective NHISs. To be effective the framework should be informed by field experiences, take into account the unique characteristics of African countries and include an open and participatory approach that encourages reuse and sharing of artifacts and experiences [10]. The framework should offer a generalized methodology and suite of tools that can be used by many countries following customization. This approach will have substantial benefits in terms of lowering the cost and risk associated with de novo implementations and "reinventing the wheel".

Desirable design features of the open architectural framework include the following:

- *Lightweight and Based on Highly Interoperable Components.* The framework must facilitate the rapid implementation of urgent, disease specific applications; integration of current components working in the field, and those developed at different times by different groups; and incorporation of carefully chosen and evolving standards [7].

- *Scalable and Extensible.* The framework must allow for incremental extension into a fully integrated NHIS without substantial reengineering and must scale to national level without requiring a fundamental change in technology or design paradigm.
- *Flexible and Adaptable.* In order to optimize chances of adoption and be deployable in different settings, the framework needs to be configurable and adaptable and easily customized to fulfill current requirements.
- *Reusable.* The framework should enhance the reusability of components to stop the wasteful cycle of developing the same functionality over and over and to achieve economies of scale.
- *Powerful, yet Easy to Use.* The framework must balance ease of use with power such that it appeals to HIS developers and genuinely facilitates the development of NHIS.

3 HEAL: The Health Enterprise Architecture Lab

The HEAL was established with seed funding from the Rockefeller Foundation and International Development Research Centre (IDRC) to undertake research in health enterprise architectures for developing African countries and to train postgraduate Computer Science students in this area.

The HEAL fills a gap in the current implementation landscape by creating a neutral space in Africa to continuously reflect on and innovate architectures and technologies for African HIS. The lab aims to develop and curate a repository of knowledge, expertise and people to develop new solutions to deal with the changing and unique circumstances and environments in African countries. Strong links and integration of the innovation process with implementing organizations ensures translation of this research into practice.

Figure 1 depicts the lab's model for conducting its main activities of research and training. The lab is hosted within the newly formed Centre for Artificial Intelligence (CAIR)[5] within the School of Computer Science at the University of KwaZulu-Natal in Durban, South Africa. It has strong links with the Knowledge Representation and Reasoning (KR&R) group based at South Africa's national ICT research facility, the CSIR Meraka Institute and the South African Medical Research Council. This provides the lab with access to academic faculty with expertise in open systems and architectures, and knowledge representation technologies, e.g. ontologies.

The lab's research is not only informed by the state of the art in Computer Science and artificial intelligence but also actual field experiences and requirements from implementation partners such as Jembi Health Systems[6], a South African based not for profit organisation. This ensures that the technologies produced by the lab are grounded by real requirements and challenges and are relevant and usable within developing African countries. Training highly skilled technical innovators is an integral part of the lab's mandate. The lab provides

[5] `http://cair.cs.ukzn.ac.za`
[6] `http://www.jembi.org`

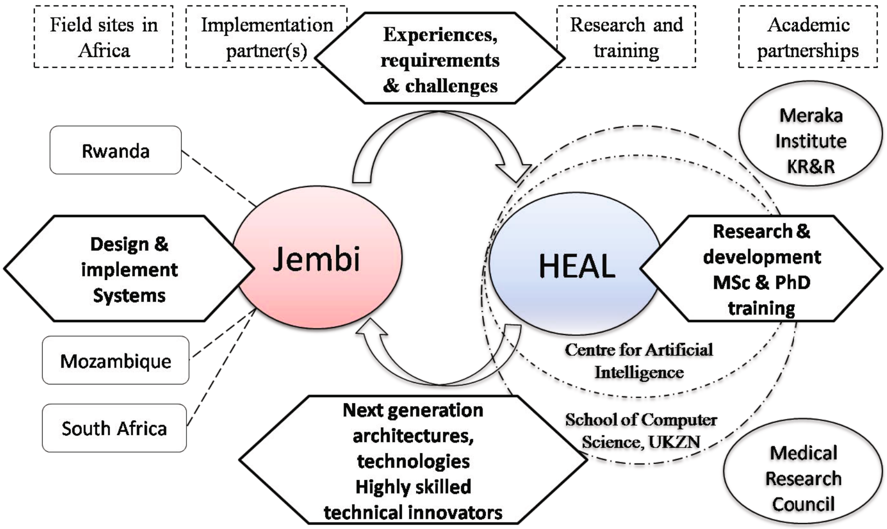

Field sites in Africa
Implementation partner(s)
Experiences, requirements & challenges
Research and training
Academic partnerships
Meraka Institute KR&R
Rwanda
Design & implement Systems
Jembi
HEAL
Research & development MSc & PhD training
Mozambique
South Africa
Centre for Artificial Intelligence
School of Computer Science, UKZN
Next generation architectures, technologies Highly skilled technical innovators
Medical Research Council

Fig. 1. The HEAL Innovation Cycle

an environment for students to undertake research masters and doctoral studies in Computer Science with a specialization in Health Informatics. This not only serves to develop new capacity in the area but also to strengthen existing capacity. The lab provides a mechanism for technical staff from implementing partners, e.g. Jembi, to obtain higher degrees and to become more effective HIS designers and implementers. This model is already demonstrating value in existing implementation and research projects. For example, the lead Jembi developer tasked with the design and implementation of an interoperability architecture for exchanging heterogenous health information in Rwanda is also working towards a research masters degree in the lab. The research involves the development of a generic interoperability framework for health information systems in developing countries. From a research perspective the Rwandan implementation provides a concrete case study that informs the requirements and evaluation of the framework. From the implementation perspective the developer is able to leverage the expertise of the lab's faculty to incorporate the latest developments and thinking in Computer Science into the Rwandan implementation.

The development of a health enterprise architectural framework (HEAF) and an architectural artifact repository (HEART) are two of HEAL's current activities towards creating an open architectural framework. HIS designers and implementers will use the HEAF and one or more artifacts from the HEART for in-country projects and feed implementation experiences back into an ongoing process of informing and refining HEAF. HEAL will continuously reflect and learn from field experiences and innovate new technologies and disseminate these to the community via the HEAF and the HEART.

4 HEAF: The Health Enterprise Architecture Framework

An Enterprise Architecture approach is increasingly being recognised as a systematic and rational approach for analysing and documenting an integrated HIS in low income countries [24].

> *Enterprise Architecture (EA) is a coherent whole of principles, methods and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems and infrastructure.* [20].

The Health Enterprise Architecture Framework (HEAF) aims to develop a set of principles and a systematic approach for modeling different aspects of a national health information systems (NHIS) specifically for low resource settings. Individual countries will use the framework to create country-specific architectures. HEAF will draw on several existing frameworks. It is a simplification of the Generic Component Model approach [3] (GCM) combined with elements from the Zachman Framework [28] applied to healthcare[7], the Federal Enterprise Architecture Framework [13], the Open Group Architectural Framework [15] the development framework for interoperable health information systems (HIS-DF) [21] and the Health Metrics Network Framework [19]. HIS-DF is a consistent application of an EA approach to the healthcare domain and addresses issues relevant to developing countries, such as the integration of public health and clinical systems [4]. HEAF constrains the GCM and HIS-DF to create a more simplified form of the HIS-DF for low resource settings. HEAF also uses a domain-specific ontology to integrate the logical layers [6] of the HEAF from the enterprise viewpoint to detailed concept models [14].

HEAF is adding domain-specific specializations, artifacts and patterns based on actual implementation experiences in South Africa, Mozambique and Rwanda as well as documented experiences in a growing number of other developing countries with architecture projects, including Ghana, Kenya and the Philippines. Although the basic principles of healthcare in low and high resource settings are similar, there are many practical differences that drive the need for a dedicated framework for low resource settings, including dedicated views and viewpoints as well as unique artifacts and patterns. Lessons learned may, in turn, inform designs and practices elaborated in the developed world, through partnerships with other applied research labs, such as the Mohawk Applied Research Centre[8] in Ontario.

A key distinguishing feature of HEAF is simplicity. HEAF is attempting to distil out the best practices and artifacts that have worked or become entrenched in several developing countries and generalize these into a framework that can be applied in other countries. The intended result is a user-oriented, practical framework that balances ease of adoption and use with completeness and theoretical rigor. HEAF is being established as a community-driven web portal[9]

[7] https://apps.adcom.uci.edu/EnterpriseArch/Zachman/Resources/ExampleHealthCareZachman.Pdf

[8] http://www.mohawkcollege.ca/about/research/marc/healthInformatics.html

[9] http://heaf.jembi.org

where users may share implementation experiences and participate in generalizing these experiences to the emergent framework. The over-arching purpose of the HEAF is to assist in developing HIS design blueprints and promoting informed decision-making by the Ministry of Health.

The expected benefits of the HEAF are to lower the cost and effort to build and extend African health information systems, to facilitate reuse, to improve interoperability and to strengthen NHIS. In this regard HEAF aims to populate a repository (HEART) with a number of well-characterized artifacts that are currently being used in African HIS. Information obtained from in-country implementations is presently being compiled and modeled. Patterns and general models are being extracted and tested against in-country implementation, consistent with our bottom-up, implementation-driven approach.

5 HEART: The Health Enterprise Architecture Repository of Tools

The HEART project aims to design a web-accessible, community driven catalogue of the various architectural artifacts that currently exist within HIS in developing countries. Examples of artifacts are: software tools and platforms; architectural designs and patterns; standards, policies and requirements.

Core to HEART is the ability to index and classify the artifacts and building blocks of a NHIS as well as a clearly defined metadata model of each of these. Formal ontologies [16] are increasingly being used for modeling and sharing domain knowledge. Ontologies have also been proposed to describe and capture the organization and characteristics of system components [17]. In such a system, termed an ontology driven information system [17], ontologies capture an online model of the system. Developers can reconfigure aspects of the system at runtime by manipulating the appropriate ontologies. We have already investigated this idea within an open geospatial architecture [22] [23] and its importance to HIS architectures has recently been highlighted [5] [6].

The first step towards building a health architecture ontology for developing countries is to conduct a survey of and model the characteristics of current architectural artifacts. The initial phase will catalogue the most widely used software artifacts deployed in African countries, such as the Open Medical Record System[10] (OpenMRS), the District Health Information System[11](DHIS) and their interoperability characteristics, including the standards they support, e.g. HL7 messaging and SDMX-HD[12] . The ontological model will be created by consultation with key role players in the community and will aim to capture unambiguous and consistent descriptions of current artifacts in the community, and how artifacts are combined in different HIS implementations. The increasing availability of open source software technologies such as OpenMRS and DHIS presents a

[10] http://www.openmrs.org
[11] http://www.dhis2.org
[12] http://www.sdmx-hd.org/

great opportunity for developing countries [2]. Even though HEART will include both open and proprietary software and technologies, our initial phase will target mostly open and free technologies (software, standards, architectures, content etc.). This also promotes our core values of facilitating interoperability, reuse and innovation. Furthermore, it is difficult to justify the expense of proprietary solutions in countries with an overwhelming disease burden and the fundamental requirement for basic clinical and public health services. However, the high level of IT skills and long term costs required for maintaining free and open-source systems cannot be ignored.

The project ultimately aims to deliver a web-based searching and visualization tool and engine for a catalogue of architectural artifacts and building blocks for HIS in developing countries. Such a catalogue will form a dynamic community resource for HIS architects and developers to share, find, compare, evaluate, and reuse artifacts. Two essential aspects of the project are to establish partnerships with providers of existing artifacts, software and functionality in order to develop a community of reviewers and contributors, and designing an ontology that adequately captures and reflects the current view of the community, but that can be easily modified to accommodate changes to this view.

6 Impact and Conclusions

In terms of their national health information systems, developing African countries differ from developed countries in that: they have limited infrastructure, have limited budgets and specialized technical skills to dedicate towards a national HIS. Foreign donors fund many African HIS implementations. There is an opportunity to pool and reuse resources across multiple countries and to optimize donor funds. This requires the creation of an architectural framework that is informed by field experiences, that takes into account the unique characteristics of African countries and an open and participatory approach that encourages reuse and sharing of artifacts and experiences. Furthermore, highly specialized local skills and capacity must be developed to ensure the future sustainability of these systems. The HEAF attempts to create a generic framework and systematic approach to ease the task of creating African national HIS. While HEAF will provide an invaluable tool for modeling and designing NHIS at the macro level, its effectiveness at the micro level for modelling loosely coupled and flexible software components is not yet clear. The HEART will provide a central point where architectural artifacts can be accessed, compared and reused. The HEAL will create a neutral space to reflect on field experiences and to continuously evolve and maintain the framework and repository and where African computer science graduates can acquire specialized skills to conduct research, maintain and innovate future African national health information systems.

Acknowledgements. The HEAL project is funded by grants from the Rockefeller Foundation (Establishing a Health Enterprise Architecture Lab, a research laboratory focused on the application of enterprise architecture and health informatics to low-resource settings, Grant Number: 2010 THS 347) and the IDRC

(Health Enterprise Architecture Laboratory (HEAL), Grant Number: 106452-001). HEAF and HEART are also funded by grants from the IDRC (Open Architectures, Standards and Information Systems (OASIS II) Developing Capacity, Sharing Knowledge and Good Principles Across eHealth in Africa. Grant Number: 105708) and Rockefeller Foundation (Open eHealth Enterprise Architecture Framework and Strategy Development for the Global South; Grant Number: 2009 THS 328).

The authors express their gratitude to the reviewers for their thoughtful comments and suggestions that helped improve the final version of the paper.

References

1. AbouZahr, C., Boerma, T.: Health information systems: the foundations of public health. Bulletin of the World Health Organization 83(8), 578–583 (2005)
2. Bagayoko, C.O., Dufour, J.C.D., Chaacho, S., Bouhaddou, O., Fieschi, M.: Open source challenges for hospital information system (HIS) in developing countries: a pilot project in Mali. BMC Med. Inform. Decis. Mak. 10(22) (2010)
3. Blobel, B.: Architectural Approach to eHealth for Enabling Paradigm Changes in Health. Methods Inf. Med. 49(2), 123–134 (2010)
4. Blobel, B.: Analysis, design and implementation of secure and interoperable distributed health information systems. IOS Press (2002)
5. Blobel, B., Kalra, D., Koehn, M., Lunn, K., Pharow, P., Rhotsalainen, P., Schulz, S., Smith, B.: The Role of Ontologies for Sustainable, Semantically Interoperable and Trustworthy EHR Solutions, pp. 953–957. IOS Press (2009)
6. Blobel, B., Oemig, F.: Ontology-Driven Health Information Systems Architectures. IOS Press (2009)
7. Braa, J., Hanseth, O., Heywood, A., Mohammed, W., Shaw, V.: Developing health information systems in developing countries: the flexible standards strategy. Management Information Systems Quarterly (2007)
8. Braa, J., Hedberg, C.: Developing District-based Health Care Information Systems: The South African Experience. In: Svensson, L., Snis, U., Srensen, C., Fgerlind, H., Lindroth, T., Magnusson, M., Stlund, C. (eds.) Proceedings of IRIS 23, Laboratorium for Interaction Technology, University of Trollhättan/Uddevalla (2000)
9. Braa, J., Hedberg, C.: The Struggle for District-Based Health Information Systems in South Africa. The Information Society 18(2), 113–127 (2002)
10. Braa, J., Monteiro, E., Sahay, S.: Networks of Action: Sustainable Health Information Systems across Developing Countries. MIS Quarterly 28(3) (2004)
11. Chan, M., Kazatchkine, M., Lob-Levyt, J., Obaid, T., Schweizer, J., Sidibe, M., Veneman, A., Yamada, T.: Meeting the Demand for Results and Accountability: A Call for Action on Health Data from Eight Global Health Agencies. PLoS Med. 7(1), e1000223+ (2010)
12. Vital Wave Consulting: Health Information Systems in Developing Countries. Tech. rep. (May 2009)
13. CIO Council: A Practical Guide to Federal Enterprise Architecture. Tech. rep. (February 2001)
14. Goossen, W., Goossen-Baremans, A., van der Zel, M.: Detailed Clinical Models: A Review. Healthc Inform. Res. 16(4), 201–214 (2010)
15. The Open Group: TOGAF Version 9. The Open Group Architecture Framework (2009), http://www.opengroup.org/architecture/togaf9/downloads.htm

16. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. Journal of Human-Computer Studies 43, 907–928 (1995)
17. Guarino, N.: Formal Ontology and Information Systems. In: Proceedings of the 1st International Conference on Formal Ontology in Information Systems, 1st edn., Trento, Italy, June 6-8. IOS Press, Amsterdam (1998)
18. Heeks, R.: Information Systems and Developing Countries: Failure, Success, and Local Improvisations. The Information Society 18(2), 101–112 (2002)
19. HMN: HMN Framework. 2nd edn., http://www.who.int/healthmetrics/documents/framework/en/index.html
20. Lankhorst, M.: Enterprise Architecture at Work: Modelling, Communication and Analysis, 2nd edn. The Enterprise Engineering Series. Springer (September 2009)
21. Lopez, D., Blobel, B.: A development framework for semantically interoperable health information systems. International Journal of Medical Informatics 78(2), 83–103 (2009)
22. Moodley, D., Tapamo, J.: A semantic infrastructure for a Knowledge Driven Sensor Web. In: Proceedings of the 4th International Workshop on Semantic Sensor Networks (SSN 2011), Bonn, Germany, October 23 (2011); A Workshop of the 10th International Semantic Web Conference (ISWC 2011)
23. Moodley, D.: Ontology Driven Multi-Agent Systems: An Architecture for Sensor Web Applications. Ph.D. thesis, School of Computer Science, University of KwaZulu-Natal (2009)
24. Mwanyika, H., Lubinski, D., Anderson, R., Chester, K., Makame, M., Steele, M., et al.: Rational Systems Design for Health Information Systems in Low-Income Countries: An Enterprise Architecture Approach. Journal of Enterprise Architecture 7(4) (2011)
25. Omary, Z., Lupiana, D., Mtenzi, F., Wu, B.: Analysis of the Challenges Affecting E-healthcare Adoption in Developing Countries: A Case of Tanzania. International Journal of Information Studies 2(1) (2010)
26. Stansfield, S., Orobaton, N., Lubinski, D., Uggowitzer, S., Mwanyika, H.: The Case for a National Health Information System Architecture; a Missing Link to Guiding National Development and Implementation (2008)
27. WHO: Everybody's Business: Strengthening Health Systems to Improve Health Outcomes: WHO's Framework for Action (2007), http://www.who.int/healthsystems/strategy/en/
28. Zachman, J.A.: A framework for information systems architecture. IBM Systems Journal 26(3), 276–292 (1987)

Formalization of Heart Models Based on the Conduction of Electrical Impulses and Cellular Automata

Dominique Méry and Neeraj Kumar Singh

Université de Lorraine
LORIA, BP 239, 54506 Vandoeuvre lès Nancy, France
{mery,singhnne}@loria.fr

Abstract. Tools and techniques based on formal methods have been recognized as a promising approach to supporting the process of verification and validation of critical systems in the early stages of their development. In particular, medical devices are very prone to showing unexpected system behaviour in operation because of the stochastic nature of the systems and when traditional methods are used for system testing. Device-related problems have been responsible for a large number of serious injuries. Officials of the US Food and Drug Administration (FDA) have found that many deaths and injuries related to these devices are caused by flaws in product design and engineering. Cardiac pacemakers and implantable cardioverter–defibrillators (ICDs) are the most critical of these medical devices, requiring closed-loop modelling (integrated system and environment modelling) for verification purposes before obtaining a certificate from the certification bodies. No technique is available to provide environment modelling for verifying the developed system models. This paper presents a methodology for modelling a biological system, such as the heart, to enable modelling in a biological environment. The heart model is based mainly on electrocardiography analysis, which models the heart system at the cellular level. The main objective of this methodology is to model the heart system and integrate it with a model of a medical device such as a cardiac pacemaker to specify a closed-loop model. To build an environment model for a closed-loop system is currently an open problem. The industry has long sought such an approach to validating a system model in a virtual biological environment. Our approach involves a pragmatic combination of formal specifications of the system and the biological environment to model a closed-loop system that enables verification of the correctness of the system and helps to improve the quality of the system.

Keywords: Heart Model, ECG, Cellular Automata, EVENT B, Proof-based development, Refinement.

1 Introduction

The human heart is well known as a mechanical device of amazing efficiency that pumps blood via the circulatory system continuously throughout the person's lifetime. It is one of the most complex and important biological systems, providing oxygen and nutrients to the body to sustain life [1]. The regular impulses generated by the heart result in rhythmic contractions through a sequence of muscles in the heart, beginning at the natural pacemaker known as the sinoatrial (SA) node, which produces an action potential that travels across the atrioventricular (AV) node, the bundle of His and the Purkinje fibres distributed throughout the ventricles. The pattern and the timing of these impulses determine the heart rhythm. Variable time intervals and conduction speeds during the heartbeat generate abnormal heart rhythms, which are also known as heart rhythm impairments. Heart rhythm impairment is the principal source of several diseases. Electrocardiography analysis is frequently used to diagnose various types of heart disease [2] by presenting the timing properties of the electrical system of the heart. These are the most fundamental properties of the heart.

Cardiac pacemakers and ICDs are the two main types among the remarkable range of medical and technological devices recommended by doctors in cases of abnormal heart rhythm. These devices are used to maintain the heart rhythm, and are life-saving in many instances. In the last few years, the use of cardiac pacemakers and cardioverter–defibrillators has increased. However, these devices may sometimes malfunction. Device-related problems have been responsible for a large number of serious injuries. Many deaths and injuries caused by device failure have been reported by the FDA [3], which advocates safety and security guidelines for using these devices. FDA officials have found that many deaths and injuries related to the devices are caused by product design and engineering flaws, which can be considered as firmware problems [4, 5].

Tools and techniques based on formal methods are considered as de facto standards for developing highly critical systems such as avionic, automotive and medical systems. Because software plays an increasingly important role in medical devices and in healthcare-related activities more generally, regulatory agencies such as the FDA and certification bodies such as the FDA's Quality System Regulation and the International Standards Organization's 13485 [6, 5, 7] need effective methods for ensuring that newly developed software-based healthcare systems are *safe* and *reliable*. Regulatory agencies, in addition to the medical device manufacturers themselves, have been striving for a more rigorous engineering-based review strategy to provide this assurance [8]. This makes formal approaches appealing. Formal-model-based methods have been successful in targeted applications of medical devices [9–12, 8, 7]. Over the past decade, there has been considerable progress in the development of formal methods for improving confidence in complex software-based systems [13, 14]. Although formal methods are part of the standard recommendations for developing and certifying medical systems, the integration of formal methods into the certification process is, in large part, unclear. In particular, it is a very challenging task to ensure that the end product of the software-development system behaves securely.

1.1 Motivation

The most challenging problem is environment modelling. That is, to validate and to verify the correct behaviour of a system model requires an interactive formal model of the environment. For example, a formal model of a cardiac pacemaker or ICD requires a heart model to verify the correctness of the developed system (see Fig. 1). No tools and techniques are available to provide environment modelling that would enable verification of the developed system model. Medical devices are tightly coupled with their biological environment (i.e., the heart) and use actuators and sensors to interact with the biological environment. Because of this strong relationship between the medical device (e.g., a pacemaker) and the related biological environment (i.e., the heart), it is necessary to model the functioning of the medical device within the biological environment. The environment model will be independent of the device model, which is helpful in creating an environment for medical devices that simulates the actual behaviour of the system. The medical device model will be dependent on the biological environment. Whenever an undesired state occurs in the biological environment, the device model must act according to the requirements. The main objective is to use a formal approach to modelling the medical device and the biological environment to verify the correctness of the medical system.

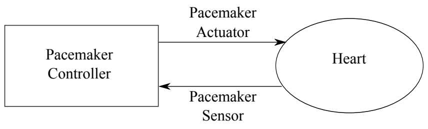


Fig. 1. Cardiac Pacemaker and Heart Interaction

To model the biological environment (the heart) for a cardiac pacemaker or ICD, we propose a method for modelling the heart using logico-mathematical theory. The heart model is based on electrocardiography analysis [15, 2, 16], which models the heart system at the cellular level [17]. In this investigation, we present a methodology for modelling a heart that involves extracting a set of biological nodes (SA node, AV node, etc.), impulse propagation speeds between nodes, impulse propagation times between nodes and cellular automata (CA) for propagating impulses at the cellular level. This model is developed through incremental refinement, which introduces several properties in an incremental way and verifies the correctness of the heart model. A key feature of this heart model is the representation of all possible morphological states of the (ECG) [16, 18]. These morphological states represent both the normal and the abnormal states of the ECG. The morphological representation can generate any kind of heart model (a patient's model or a normal heart model) using the ECG. This model can observe both the failure of impulse generation and the failure of impulse propagation. The mathematical heart model, based on logico-mathematical theory, is verified using the RODIN [19] proof tool and the model checker ProB. The model is also verified by electro-physiology and cardiac experts. The main objective of this heart model is to provide a biological environment (the heart) for formalizing a closed-loop system (a combined model of a cardiac pacemaker and the heart).

1.2 Outline of the Paper

An outline of the remainder of this paper is as follows. Section 2 presents related work. A brief outline of the heart system is introduced in Section 3. Section 4 explains the proposed approach. Section 5 gives an outline of the formal development of the heart model. Section 6 discusses the results of lessons learned from this experience, and Section 7 concludes the paper with some perspectives together with proposals for future work.

2 Related Work

Heart modelling is a challenging problem in the area of real-time simulation for clinical purposes. It is handled by the research community using a variety of different methods. The ECG is an important diagnostic method for measuring the heart's electrical activities, and was invented by Willem Einthoven in 1903 [20]. In this study, the ECG is used in modelling the heart [20]. At the present time, technological advances have enabled the production of a high-quality cellular model of an entire heart.

K. R. Jun et al. [21] have produced a CA model of the activation process in ventricular muscle tissue. They presented a two-dimensional (2D) CA model that accounts for the local orientation of the myocardial fibres and their distributed velocity and refractory period. A three-dimensional (3D) finite-volume-based computer mesh model of human atrial activation and current flow has been presented by Harrild et al. [15]. This cellular-level-based model included both the left and right atria and the major muscle bundles of the atria. The results of using this model demonstrate a normal sinus rhythm and can extract the patterns of the septum's activation. Because of memory and time complexity in the computation of a 3D model, an empirical approach is used in modelling the whole heart. The empirical approach implies a simpler representation of the complex process at a cellular level. In this new approach, researchers have adopted some approximations in modelling the whole heart without compromising the actual behaviour of the heart. Berenfeld et al. [22] have developed a model that can give insight into the local and global complex dynamics of the heart in the transition from normal to abnormal myocardial activity, which helps to estimate myocardial properties. Adam [23] has analysed wave activities during depolarization in his cardiac model, which is represented by a simplification of the heart tissue.

Recently, a real-time Virtual Heart Model (VHM) has been developed by Jiang et al. [24] to model the electro-physiological operation of proper functioning and malfunctioning. They used a time-automaton model to define the timing properties of the heart. Simulink Design Verifier[1] was used as the main tool for designing the VHM.

Our approach is based purely on formal techniques for modelling the heart using electrocardiography analysis. To model the heart for a cardiac pacemaker or ICD, we propose a method based on logico-mathematical theory, which can be implemented using any formal-methods-based tools (Z, TLA$^+$, VDM, etc.). In this paper, the model is developed using a maximal refinement approach at the cellular level. The incremental refinement approach helps both to introduce several properties in an incremental way and to verify the correctness of the heart model. The key feature of this heart model is the representation of all possible morphological states of the ECG, which is used to represent both normal and abnormal states through observation of the failure of impulse generation and the failure of impulse propagation in the heart [16, 2, 1, 18].

[1] `http://www.mathworks.com/products/sldesignverifier/`

3 Background

3.1 The Heart System

The human heart is wondrous in its ability to pump blood to the circulatory system continuously throughout a lifetime. The heart comprises four chambers: right atrium, right ventricle, left atrium and left ventricle, each of which contract and relax periodically. The atria form one unit and the ventricles another. The heart's mechanical system (the pump) requires impulses from its electrical system to function. An electrical stimulus is generated by the sinus node (see Fig. 2), which is a small mass of specialized tissue located in the right atrium of the heart.

The electrical stimulus travels down through the conduction pathways and causes the heart's lower chambers to contract and pump out the blood. The right and left atria are stimulated first and contract for a short period of time before the right and left ventricles. Each contraction of the ventricles represents one heartbeat. The atria contract for a fraction of a second before the ventricles, so their blood empties into the ventricles before the ventricles contract.

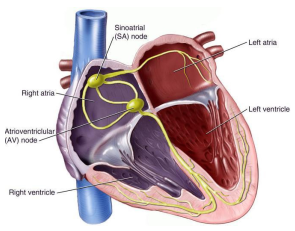


Fig. 2. The Heart

Arrhythmias are caused by cardiac problems that produce abnormal heart rhythms. In general, arrhythmias reduce haemodynamic performance, including situations where the heart develops an abnormal rate or rhythm or when normal conduction pathways are interrupted, and a different part of the heart takes over control of the rhythm. An arrhythmia can involve an abnormal rhythm increase (tachycardia: $>$ 100 bpm) or decrease (bradycardia: $<$ 60 bpm), or it may be characterized by an irregular cardiac rhythm, such as that caused by asynchrony of the cardiac chambers. Irregularities in the heartbeat are called bradycardia and tachycardia. Bradycardia indicates that the heart rate falls below the expected level whereas tachycardia indicates that the heart rate goes above the expected heart rate. An artificial pacemaker can restore synchrony between the atria and the ventricles [25–29, 1]. Beats per minute (bpm) is the basic unit used to measure the rate of heart activity.

3.2 Overview of the ECG

The ECG (or EKG) [2, 27, 16, 18] is a diagnostic tool that measures and records precisely the electrical activity of the heart in the form of signals. Clinicians can evaluate the condition of a patient's heart from the ECG and perform further diagnoses. Analysis of these signals can be used to diagnose a wide range of heart conditions and to predict the related diseases. ECG records are obtained by sampling the bioelectric currents sensed by several electrodes, known as leads. A normal ECG is depicted in Fig. 3. All the segments and intervals used by clinicians are represented in this ECG diagram. Depolarization and repolarization of the ventricular and atrial chambers are presented by deflection in the ECG signal. These deflections are labelled in alphabetic order: P-QRS-T. Sequential activation, depolarization and repolarization are distinct deflections in the ECG, caused by anatomical differences between the atria and the ventricles. The sequences are even distinguishable when they are not in the correct sequence (P-QRS-T). Each beat of the heart can be observed as a series of deflections, which reflects the time evolution of electrical activity in the heart [16, 2, 16, 18]. A single cycle of the ECG is considered as one heartbeat. The ECG may be divided into the following sections.

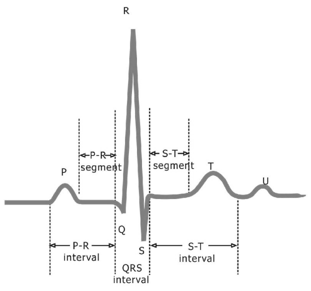


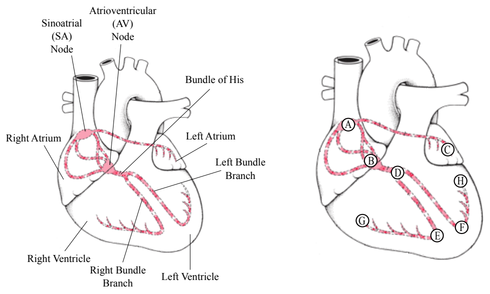
Fig. 3. A Typical ECG Tracing

- **P-Wave:** A small low-voltage deflection caused by the depolarization of the atria prior to atrial contraction as the activation (depolarization) wave front propagates from the SA node through the atria.
- **PQ-Interval:** The time between the beginning of atrial depolarization and the beginning of ventricular depolarization.
- **QRS-Complex:** The QRS-complex is easily identifiable between the P- and T-waves because it has a characteristic waveform and dominating amplitude. The dominating amplitude is caused by currents generated when the ventricles depolarize prior to their contraction. Although atrial repolarization occurs before ventricular depolarization, the latter waveform (i.e., the QRS-complex) is of much greater amplitude, and atrial repolarization is therefore not seen on the ECG.
- **QT-Interval:** The time between the onset of ventricular depolarization and the end of ventricular repolarization. Clinical studies have demonstrated that the QT-interval increases linearly as the RR-interval increases [4]. A prolonged QT-interval may be associated with delayed ventricular repolarization, which may cause ventricular tachyarrhythmias leading to sudden cardiac death [9].
- **ST-Interval:** The time between the end of the S-wave and the beginning of the T-wave. Significantly elevated or depressed amplitudes away from the baseline are often associated with cardiac illness.
- **T-Wave:** Ventricular repolarization, whereby the cardiac muscle is prepared for the next cycle of the ECG.

4 Proposed Idea

Our proposed method exploits a heart model based on logico-mathematics to help the formal methods community to verify the correctness of a developed model of medical devices such as cardiac pacemakers. The heart model is based mainly on the impulse propagation time and conduction speed at a cellular level. This method uses the advanced capabilities of a combined approach of formal verification and model validation using a model checker to achieve considerable advantages in heart system modelling.

Fig. 4(a) shows the more significant components and the impulse conduction path in the entire heart system. The heart is a muscle with a special electrical conduction system. The system comprises two nodes (special conduction cells) and a series of conduction fibres or bundles (pathways). For modelling the heart system, we have assumed eight landmark nodes (A, B, C, D, E, F, G, H) in the whole conduction network, as shown in Fig. 4(b), which control the whole heart system. These landmarks were identified via a literature survey [1, 2, 16, 18] and extensive discussions with a cardiologist and a physiologist.



(a) Basic Electrical Conduction System (b) Landmarks in the Network

Fig. 4. Electrical Conduction and Landmarks in the Heart System

We now introduce the necessary elements we use to define the heart system formally.

Definition 1 (The Heart System). *Given a set of nodes N, a transition (conduction) t is a pair (i, j), with i, j* $\in N$ *. A transition is denoted by i* $\rightsquigarrow$ *j. The heart system is a tuple HSys = (N, T,* N_0, TW_{time}, CW_{speed} *) where:*

• *N =* $\{$ *A, B, C, D, E, F, G, H* $\}$ *is a finite set of landmark nodes in the conduction pathways of the heart system;*
• $T \subseteq N \times N = \{A \mapsto B, A \mapsto C, B \mapsto D, D \mapsto E, D \mapsto F, E \mapsto G, F \mapsto H\}$ *is a set of transitions to represent electrical impulse propagation between two landmark nodes;*
• N_0 *= A is the initial landmark node (SA node);*
• $TW_{time} \in N \rightarrow TIME$ *is a weight function for the time delay of each node, where* $TIME$ *is a range of time delays;*
• $CW_{speed} \in T \rightarrow SPEED$ *is a weight function for the impulse propagation speed of each transition, where* $SPEED$ *is a range of propagation speeds.*

Property 1 (Impulse Propagation Time). *In the heart system, the electrical impulse originates from the SA node (node A), travels through the entire conduction network and terminates at the atrial muscle fibres (node C) and at the ends of the Purkinje fibres in both sides of the ventricular chambers (node G and node H). The impulse propagation time delay differs for each landmark node (N). The impulse propagation time is represented as the total function* $TW_{time} \in N \rightarrow \mathbb{P}(0..230)$*. The impulse* *propagation time delay for each node (N) is represented as:* $TW_{time}(A) = 0..10$, $TW_{time}(B) = 50..70$, $TW_{time}(C) = 70..90$, $TW_{time}(D) = 125..160$, $TW_{time}(E) = 145..180$, $TW_{time}(F) = 145..180$, $TW_{time}(G) = 150..210$ *and* $TW_{time}(h) = 150..230$.

Property 2 (Impulse Propagation Speed). *The impulse propagation speed also differs for each transition ($i \rightsquigarrow j$, where $i, j \in N$). The impulse propagation speed is represented as the total function* $CW_{speed} \in T \to \mathbb{P}(5..400)$. *The impulse propagation speed for each transition is represented as:* $CW_{speed}(A \mapsto B) = 30..50$, $CW_{speed}(A \mapsto C) = 30..50$, $CW_{speed}(B \mapsto D) = 100..200$, $CW_{speed}(D \mapsto E) = 100..200$, $CW_{speed}(E \mapsto G) = 300..400$ *and* $CW_{speed}(F \mapsto H) = 300..400$.

Electrical activity is spontaneously generated by the SA node, located high in the right atrium, shown as node A in Fig. 5(a). The SA node is the physiological pacemaker of the normal heart, responsible for setting its rate and rhythm. The electrical impulse spreads through the walls of the atria, causing them to contract. The conduction of the electrical impulse throughout the left and right atria is seen on the ECG as the P-wave (see Fig. 3). From the sinus node, the electrical impulse propagates throughout the atria and reaches nodes B and C, but cannot propagate directly across the boundary between the atria and ventricles. The electrical impulse travels outward into the atrial muscle fibres and reaches the end of the fibres, shown as node C in the conduction network (see Fig. 5(b)).

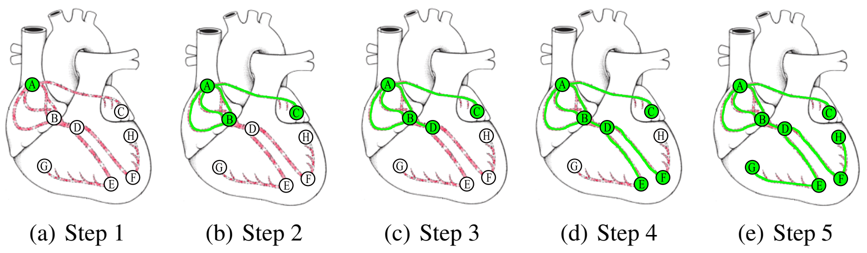


Fig. 5. Impulse Propagation through Landmark Nodes

Normally, the only pathway available for the electrical impulse is to enter the ventricles through a specialized region of cells called the AV node. The AV node is located at the boundary between the atria and ventricles, shown as node B in Fig. 4(b). The AV node provides the only conducting path from the atria to the ventricles. The AV node functions as a critical delay in the conduction system. Without this delay, the atria and ventricles would contract at the same time, and blood would not flow effectively from the atria to the ventricles. The delay in the AV node forms much of the PR segment on the ECG. Part of the atrial repolarization can be represented by the PR segment (see Fig. 3).

Propagation from the AV node (A) to the ventricles is provided by a specialized conduction system. The distal portion of the AV node is composed of a common bundle called the Bundle of His, shown as landmark node D in Fig. 4(b). The Bundle of His splits into two branches in the inter-ventricular septum, namely the left bundle branch and the right bundle branch. The electrical impulses then enter the base of the ventricle

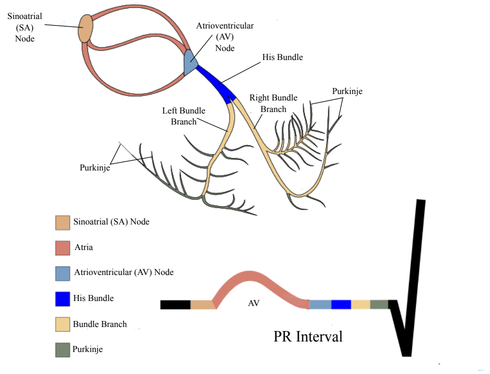


Fig. 6. Time Intervals and Impulse Propagation in the ECG Signal [1]

at the Bundle of His (node D) and follow the left and right bundle branches along the inter-ventricular septum (see Fig. 5(c)).

The two separate bundle branches propagating along each side of the septum constitute the left and right bundle branches. We have identified two landmark nodes E and F (see Fig. 4(b)) in the lower part of the heart for the left and right bundle branches. These specialized fibres conduct the impulses at a very rapid velocity (see Table 1). The left bundle branch activates the left ventricle, whereas the right bundle branch activates the right ventricle (see Fig. 5(d)).

Table 1. Cardiac Activation Time and Cardiac Velocity [1]

Location in the heart	**Cardiac Activation Time (ms)**	**Location in the heart**	**Conduction Velocity (cm/s)**
SA Node (A)	0..10	$A \mapsto B$	30..50
Left atrium muscle fibres (C)	70..90	$A \mapsto C$	30..50
AV Node (B)	50..70	$B \mapsto D$	100..200
Bundle of His (D)	125..160	$D \mapsto E$	100..200
Right Bundle Branch (E)	145..180	$D \mapsto F$	100..200
Left Bundle Branch (F)	145..180	$E \mapsto G$	300..400
Right Purkinje fibres (G)	150..210	$F \mapsto H$	300..400
Left Purkinje fibres (H)	150..230		

The bundle branches then divide into an extensive system of Purkinje fibres that conduct the impulses at high velocity (see Table 1) throughout the ventricles. The Purkinje fibres stimulate individual groups of myocardial cells to contract. We have identified two final landmark nodes G and H (see Fig. 4(b)) at the end of the Purkinje fibres in both sides of the ventricles. These two nodes represent the end of the conduction network in the heart system. The bundles branch into the Purkinje fibres that diverge across the inner sides of the ventricular walls (see Fig. 5(e)). On reaching the end of the Purkinje fibres, the electrical impulse is transmitted through the ventricular muscle mass by the ventricular muscle fibres themselves. Propagation along the conduction system takes place at a relatively high speed once it is within the ventricular region, but prior to this (through the AV node), the velocity is extremely slow [1, 2].

The electrical system provides a synchronized system from atria to ventricles, which aids the contraction of the heart muscle and optimizes the haemodynamics. Changed time intervals or conducting speeds between landmarks (see Fig. 4(b) and Fig. 6) are a major cause of abnormalities in the heart system. Abnormalities in electrical signals in the heart can generate various kinds of arrhythmias. A slow conduction speed generates bradycardia and a fast conduction speed generates tachycardia. In this model, we consider the ranges of all possible values for conduction speeds and conduction times for each landmark node and conduction path. This model represents the morphological structure of the ECG signal through the conduction network (see Fig. 6).

4.1 Heart Block

In this section, we explain the basic heart blocks in the heart conduction system. We have formalized these basic heart blocks in the proposed methodology. Heart block is the term given to a disorder of conduction of the impulse that stimulates heart muscle contraction. The normal cardiac impulse arises in the SA node (A), situated in the right atrium, and spreads to the AV node (B), whence it is conducted by specialized tissue known as the Bundle of His (D), which divides into the left and right bundle branches in the ventricles (see Fig. 4(a)). Disturbances in conduction may appear as slow conduction, intermittent conduction failure or complete conduction failure. These three kinds of conduction failure are also known as 1st, 2nd and 3rd degree blocks. We can show these different kinds of heart block throughout the conduction network in terms of our set of landmark nodes (see Fig. 7).

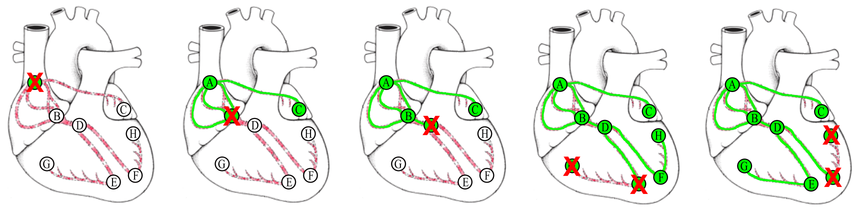
SA Block. This block occurs within the SA node (A) and is described as an SA nodal block or sick sinus syndrome. The SA node fails to originate an impulse, and the heart misses one or two beats at regular or irregular intervals (see Fig. 7(a)).

AV Block. For an AV block, the sinus rhythm is normal, but there is a conduction defect between the atria and the ventricles. The main cause of this block may be in the AV node (B) or the Bundle of His (D), or both (see Fig. 7(b)).

Infra-Hisian Block. Blocks that occur below the AV node (B) are known as Infra-Hisian blocks (see Fig. 7(c)).

Left Bundle Branch Block. In the normal heart, activation of both ventricles takes place simultaneously. A left bundle branch block occurs when conduction into the left branch of the Bundle of His is interrupted. Blocks that occur within the fascicles of the left bundle branch are known as hemiblocks (see Fig. 7(d)).

Right Bundle Branch Block. A right bundle branch block occurs when conduction into the right branch of the Bundle of His is interrupted (see Fig. 7(e)).



(a) SA Block (b) AV Block (c) Infra-Hisian (d) RBBB (e) LBBB

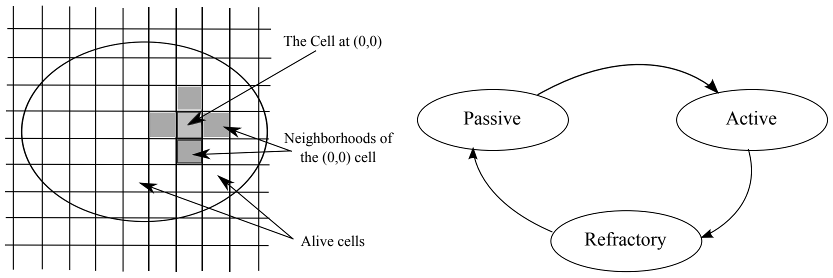
Fig. 7. Impairments in Impulse Propagation Caused by Heart Blocks

4.2 CA Model

A set of spatially distributed cells form a CA model, which contains a uniform connection pattern among neighbouring cells and local computation laws. CA were originally proposed by Ulam and von Neumann in the 1940s to provide a formal framework for investigating the behaviour of complex, spatially distributed systems [17]. CA are discrete dynamic systems corresponding to space and time. CA modelling involves uniform properties for state transitions and interconnection patterns. The model components are specified by a single property caused by the same patterns instead of specifying each component separately. CA models help to visualize a system's dynamics [30, 15, 1]. A CA model can have an infinite number of cells along any dimension. Here, we consider a finite number of cells in two dimensions, as shown in Fig. 8(a). A 2D CA model is defined as:

Definition 2 (The CA Model)
$CA = < S, N, T >$: discrete time system
S : the set of states
N: the neighbouring patterns at (0,0),
T: the transition function
In the usual case of CA realized on a D-dimensional grid, N comprises D-tuples of indices from a coordinate set:
I: $N \subseteq I^D$.
The 2D cellular model therefore becomes
$N \subseteq I^2$,
$T : S^{|N|} \rightarrow S$.
To consider an automaton specified as a CA, let λ and α be the global state and the global transition function of the CA, respectively. Then, $\lambda = \{\tau | \tau : I^2 \rightarrow S\}$ and $\alpha(\lambda(i, j)) = T(\tau | N + (i, j))$ for all τ in λ and (i,j) in I^2.

Definition 3 (State Transition of a Cell). *The heart muscle system is composed of heterogeneous cells, but the CA model of the muscle system, CAM_{CA}, is characterized by having no dependency on the type of cells. CAM_{CA} is defined as follows:*
$CAM_{CA} = < S, N, T >$
$S = \{Active, Passive, Refractory\}$
$N = \{(0, 0), (1, 0), (-1, 0), (0, 1), (0, -1)\}$



(a) A 2D CA Model (b) State Transition of a Cell

Fig. 8. The 2D CA and State Transition Model

$s'_{m,n} = s_{m,n}(t+1)$
$s'_{m,n} = T(s_{m,n}, s_{m+1,n}, s_{m-1,n}, s_{m,n+1}, s_{m,n-1})$,
where $s_{m,n}$ denotes the state of the cell located at (m,n) and T is a transition function for CAM_{CA} that specifies the next state, as shown in Fig. 8(b).

Each cell in the heart muscle should be in one of the states $Active$, $Passive$ or $Refractory$. Initially, all cells are $Passive$. In this state, the cell is discharged electrically and has no influence on its neighbouring cells. When an electrical impulse propagates, the cell becomes charged and eventually activated ($Active$ state). The cell then transmits an electrical impulse to its neighbour cells. The electrical impulse is propagated to all the cells in the heart muscle. After activation, the cell becomes discharged and enters the $Refractory$ state within which the cell cannot be reactivated. After a time, the cell changes its state to the $Passive$ state to await the next impulse.

5 Formalization of the Heart Model

To formalize the heart model, we have used the EVENT B modelling language [19, 13], although the proposed idea can be formalized using any kind of formal-methods tool such as Z, ASM, TLA^{+} or VDM. The EVENT B modelling language supports the refinement approach [31] that helps to verify the correctness of the system in an incremental way. The heart model development is expressed in an abstract and general way. The initial model formalizes the system requirements and environmental assumptions, whereas the subsequent models introduce design decisions for the resulting system.

5.1 The Context and Abstract Model

EVENT B models are described in terms of two major components: *context* and *machine*. The context contains the static part of the model, whereas the machine contains the dynamic part. The context uses sets and constants to define axioms and theorems. Axioms and theorems represent the logical theory of the elements of a system. The logical theory lists the static properties of constants related to the system and provides an axiomatization of the system environment. The context can be extended by other contexts and referenced by a set of machines, while a machine can be refined by other machines.

We need to choose electrical features for modelling the heart system. To model the heart system, we identify a set of electrical impulse propagation nodes *ConductionNode* of the heart conduction network (see Fig. 4(a)). These nodes are basic landmarks, which enable expression of the normal and abnormal behaviour of the heart system. These landmarks were identified through a literature survey [1, 2, 16, 18] and fruitful discussions with a cardiologist and a physiologist. Three constants define the impulse propagation time, namely *ConductionTime*, impulse propagation path *ConductionPath* and impulse propagation velocity *ConductionSpeed*. Static properties are defined in the context model to specify the electrical impulse propagation network of the heart system, the impulse propagation time for each landmark node and the impulse propagation speed for every path. Paths are represented by a set of pairs of landmark nodes (see Definition 1, Properties 1 and 2 and Table 1).

$$\begin{array}{l}
axm1 : partition(ConductionNode, \{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{F\}, \{G\}, \{H\}) \\
axm2 : ConductionTime \in ConductionNode \rightarrow \mathbb{P}(0\,..\,230) \\
axm3 : ConductionPath \subseteq ConductionNode \times ConductionNode \\
axm4 : ConductionSpeed \in ConductionPath \rightarrow \mathbb{P}(5\,..\,400)
\end{array}$$

From the above, axioms are extracted from the definitions, which are validated by the cardiologist and physiologist. We define an abstract model for indicating the heart state according to observations of impulse propagation in the conduction nodes. The machine model represents the dynamic behaviour of the heart system via stepwise impulse propagation into the atrial and ventricular chambers. To define the dynamic properties, we introduce four variables, namely *ConductionNodeState*, *CConductionTime*, *CConductionSpeed* and *HeartState*, using a set of invariants. The variable *ConductionNodeState* is defined as a function that shows the Boolean state of a landmark node. When an electrical impulse passes through landmark nodes (see Fig. 4(b)), the visited nodes become *TRUE* and the unvisited landmark nodes are represented by *FALSE*. The variables *CConductionTime* and *CConductionSpeed* represent the current impulse propagation time and velocity in the conduction network. The last variable *HeartState* can have Boolean states *TRUE* or *FALSE*. *TRUE* represents the normal condition of the heart, whereas *FALSE* represents an abnormal condition of the heart.

$$\begin{array}{l}
inv1 : ConductionNodeState \in ConductionNode \rightarrow BOOL \\
inv2 : CConductionTime \in ConductionNode \rightarrow 0\,..\,300 \\
inv3 : CConductionSpeed \in ConductionPath \rightarrow 0\,..\,500 \\
inv4 : HeartState \in BOOL
\end{array}$$

In the abstract specification of the heart model, there are three events, namely *HeartOK* to represent a normal state of the heart, *HeartKO* to express an abnormal state of the heart and *HeartConduction* to update the value of each landmark node in the conduction network in terms of visited landmark nodes (*ConductionNodeState*), impulse propagation intervals (*CConductionTime*) and impulse propagation velocities (*CConductionSpeed*).

The event *HeartOK* specifies a set of required conditions for the normal state of the heart system. The first guard (*grd1*) states that all landmark nodes should be visited in a single cycle of impulse propagation. The second guard states that the current impulse propagation time of each landmark node should lie within its pre-specified range of impulse propagation times. The final guard states that the current impulse propagation velocity of each path should lie between pre-defined impulse propagation velocities. If all guards are satisfied, then the heart state indicates the normal condition as being *TRUE*.

```
EVENT HeartOK
 WHEN
  grd1 : $\forall i \cdot i \in ConductionNode \Rightarrow ConductionNodeState(i) = TRUE$
  grd2 : $\forall i \cdot i \in ConductionNode \Rightarrow CConductionTime(i) \in ConductionTime(i)$
  grd3 : $\forall i, j \cdot i \mapsto j \in ConductionPath \Rightarrow$
         $CConductionSpeed(i \mapsto j) \in ConductionSpeed(i \mapsto j)$
 THEN
  act1 : $HeartState := TRUE$
 END
```

The event *HeartKO* specifies an opposite set of guards to those for the normal state of the heart system to specify abnormal conditions of the heart. These guards state that if any landmark node is not visited in a single cycle of the impulse propagation, or if the current impulse propagation time of any landmark node does not lie within the pre-specified range of impulse propagation times, or if the current impulse propagation velocity of any path does not lie within the pre-defined range of impulse propagation velocities, then the heart system is in an abnormal state, represented by its normal condition being *FALSE*. Different kinds of heart disease affect the electrical impulse propagation time and velocity in the heart system [2]. These changes affect the actual heart rhythm and help to identify the possible abnormal behaviours of the heart.

```
EVENT HeartKO
 WHEN
  grd1 : $\exists i \cdot i \in ConductionNode \land ConductionNodeState(i) = FALSE)$
         $\lor$
         $(\exists j \cdot j \in ConductionNode \land CConductionTime(j) \notin ConductionTime(j))$
         $\lor$
         $(\exists m, n \cdot m \mapsto n \in ConductionPath \land CConductionSpeed(m \mapsto n)$
         $\notin ConductionSpeed(m \mapsto n))$
 THEN
  act1 : $HeartState := FALSE$
 END
```

The event *HeartConduction* formalizes the heart behaviour in an abstract manner by updating the values for impulse propagation time, impulse propagation velocity and visited state of the landmark nodes nondeterministically. This event is used to model more concrete behaviour of the heart system at the next level of refinement.

```
EVENT HeartConduction
 BEGIN
   act1 : $ConductionNodeState :\in ConductionNode \rightarrow BOOL$
   act2 : $CConductionTime :\in ConductionNode \rightarrow 0\,..\,300$
   act3 : $CConductionSpeed :\in ConductionPath \rightarrow 0\,..\,500$
   act4 : $HeartState :\in BOOL$
 END
```

5.2 Overview of the Full Refinement Chain

So far, we have described our abstract model of the heart. Each refinement level is used to introduce a new set of functional properties for modelling normal and abnormal behaviour of the heart system. Rather than presenting the chain of refinement stages in great detail, we will just give an overview of the remaining refinement stages, sufficient to explain the rationale of each refinement stage in formalizing the heart model. For more detailed information, see the technical report [32].

Refinement 1: Introducing Steps in the Propagation. This refinement involves a conduction model of the heart, which specifies the beginning of the impulse propagation at the SA node and its end at the Purkinje fibres in the left and right ventricles. The refinement expresses the step-by-step impulse propagation through all landmark nodes, where the electrical impulse must pass through a number of intermediate landmark nodes before reaching the terminal nodes (C, G, H).

Refinement 2: Impulse Propagation. This refinement specifies impulse propagation between landmark nodes using a global clock counter to model the real-time system. This aims to satisfy the temporal properties of impulse propagation, where several events are introduced to simulate the impulse propagation in the heart conduction network. The new events involve formalizing the impulse flow between two landmark nodes individually, such as the electrical impulse movement from the SA node (A) to the AV node (B). This refinement also introduces a logical clock to synchronize all states of the heart system and check the heart states in a required period of time in the conduction network.

Refinement 3: Perturbations in Conduction. This is a perturbation model of the heart, which specifies perturbations in the heart conduction system, aiming to identify the exact block in the heart conduction system. This refinement introduces a set of possible blocks in the heart conducting system. These blocks can occur in the conduction network and affect electrical impulse propagation. Sets of landmark nodes partition the various regions for all possible heart blocks.

Refinement 4: Obtaining a Cellular Model. This is a simulation model of the heart, which introduces impulse propagation at the cellular level using CA. Cellular-level modelling is used to model the electrical impulse propagation at the cell level. The formalization uses CA theory to model the microstructure-based cell model. To formalize the CA, we introduce mathematical properties (see Definitions 2 and 3) in the context model.

Here, we have described only a summary of each refinement in the form of a very basic description of the heart-modelling incremental-refinement-based approach and have omitted the detailed formalization of events and proof details because of limited space. More detailed information about the developed formal model of the heart is given in the technical report [32].

5.3 Model Validation and Analysis

This section validates the model by using the ProB tool and proof statistics. “Validation” refers to the activity of gaining confidence that the developed formal models are consistent with the requirements. We used the ProB tool that supports *automated consistency checking* of EVENT B machines via model checking [33] and constraint-based checking. This tool assists in validating the heart model according to the conduction network and set of landmark nodes. The heart model is carefully verified through animations and under the supervision of a physiologist and a cardiologist. We have validated various scenarios of normal and abnormal heart conditions, and we have also tested the morphological behaviour [16, 18] of the ECG during impulse propagation from the SA node (A) to the Purkinje fibres (F, H) in the ventricles. The logic-based mathematical model of the heart can generate all possible scenarios of normal and abnormal heart conditions in the ECG caused by changes in time and velocity among the landmark nodes. ProB was very useful in animating all models and in verifying the absence of error (no counter-examples exist) and deadlock.

Table 2. Proof Statistics

Model	Total number of POs	Automatic Proof	Interactive Proof
Abstract Model	29	22(76%)	7(24%)
First Refinement	9	6(67%)	3(33%)
Second Refinement	159	155(97%)	4(3%)
Third Refinement	10	1(10%)	9(90%)
Fourth Refinement	11	10(91%)	1(9%)
Total	218	194(89%)	24(11%)

Table 2 expresses the proof statistics of the development using the RODIN tool. These statistics measure the size of the model, the proof obligations (POs) generated and discharged by the RODIN prover and those that are interactively proved. The complete development of the heart model results in 218 (100%) POs, within which 194 (89%) are proved automatically by the RODIN tool. The remaining 24 (11%) POs are proved interactively using the RODIN tool. For the heart model, many POs are generated because of the introduction of the new functional behaviours. To guarantee the correctness of these functional behaviours, we have established various invariants in the incremental refinements. Most of the proofs are interactively discharged in the third refinement of the heart model. These proofs are quite simple, and have been discharged with the help of simplifying predicates. Few POs are proved interactively in other refinements. This incremental refinement of the heart system helps to achieve a high degree of automatic proof.

6 Discussion

This paper presents a methodology for modelling a biological system, such as the heart, by modelling a biological environment. The main objective of this methodology is to model the heart system and integrate it with the model of a medical device such as a cardiac pacemaker, thereby modelling the closed-loop system to enable certification of the medical system via the certification bodies [6, 5] for safe operation. To build a closed-loop model using both environment and device modelling is considered a standard approach to validation, given that designing an environment model is a challenging problem in the real world. Industry has long sought such an approach to validating system models in a biological environment. We have discovered much information via a literature survey and long discussions with experts in cardiology and physiology, and have concluded how best to model the heart system as a cellular-level architecture in an efficient and optimum way. Because of the complexity of the cellular-level calculations (see Sec. 2), previous models have failed to model the heart system.

We have proposed modelling the heart in an abstract way to simulate the desired behaviour of the heart system while avoiding the complexity. More importantly, the heart model is based on logico-mathematical theory. Our primary objective was to model the heart system using only simple logico-mathematical methods. The heart model is an environmental model for medical devices that may improve their development in the early phases. As such, it will contribute only one element of the verification process. Other verification steps will also be required. Medical experts have elaborated every minor detail in an effort to understand the complexity of the biological system, particularly because the heart system is the most complex organ in the body. The proposed approach contains only a main part of the specification of the system behaviour, with the remaining information being hidden. We have spent much time identifying an exact abstract model of the heart system that satisfies medical experts. We have used the EVENT B modelling language to model and verify the system. The ProB model checker was used to verify the correctness of the heart model via animation. Any other formal specification language and model checker could be used to model the heart system based on our proposed methodology.

7 Conclusion and Future Challenges

7.1 Conclusion

This paper has presented a methodology for producing a mathematical model of a heart based on logico-mathematical theory. This model is the first computational model that considers the heart as an electrical conduction system. Given that a cardiac pacemaker interacts with the heart exactly at this level (i.e., electrical impulses), this model is a very promising "environmental model" to be used in parallel with a pacemaker model to form a closed-loop system. This model therefore has an immediate use in "the grand challenges in formal methods" where an industrial pacemaker specification has been elected as a benchmark. To formalize the heart system, we have used the EVENT B modelling language [19, 13] to develop the proof-based formal model. Our approach involves formalizing and reasoning about impulse propagation in the whole heart system through the conduction network (see Fig. 4(a)). More precisely, we would like to stress the original contribution of our work. We have proposed a method for modelling a human heart based on logico-mathematical theory. The main objectives of this proposed idea are as follows:

- To obtain a certification procedure for providing a higher safety integrity level
- To verify the system in a patient model (in a formal representation)
- To analyse the biological environment (the heart) in a mathematical way
- To analyse the interaction between the heart model and a cardiac pacemaker or ICD.

In summary, we have formalized the known characteristics and physiological behaviour of the heart. The formalization highlights various aspects of the problem, making different assumptions about impulse propagation and establishing different properties related to the CA. We have outlined how an incremental refinement approach to the heart system enables a high degree of automatic proof using the RODIN tool. Our various developments reflect not only many facets of the problem, but also the learning process involved in understanding the problem and its ultimate possible solutions.

The consistency of our specification has been checked through reasoning, and validation experiments were performed using the ProB model checker with respect to safety conditions. As part of our reasoning, we have proved that the initialization of the system is valid, and we have calculated the preconditions for operations. These have been executed to guarantee that our intention to have total operations has been fulfilled. At each stage of the refinement, we have introduced a new behaviour for the system and proved its *consistency* and performed *refinement checking*. We have introduced more general invariants at the refinement level, showing that the initialization of the whole system is valid. Finally, we have validated the heart system using the ProB model checker as a validation tool and have verified the correctness of the exact behaviour of our heart system with the help of physiology and cardiology experts.

7.2 Future Challenges

Our most important goal is that this formal model helps to obtain certification for medical devices related to the heart system, such as cardiac pacemakers and ICDs. It can be also used as a diagnostic tool to diagnose disease with the help of a patient model. This has been a first attempt at heart modelling based on logico-mathematical theory. We have successfully modelled electrical impulse propagation in the heart system. A main cause of many heart diseases is problems in the heart conduction network [2, 16]. Medical devices are tightly coupled with their biological environment (i.e., the heart), and use actuators and sensors to respond to the biological environment. Because of the strong relationship between medical devices (such as pacemakers) and their related biological environment (the heart), it is necessary to model the functioning of the medical device within the biological environment. The environment model is independent of the device model, which helps in creating an environment for the medical device that simulates the desired behaviour of a device. The medical device model is dependent on the biological environment. Whenever any undesired state occurs in the biological environment, the device model must act according to the requirements. In future work, our main objective will be to integrate the formal specification of a pacemaker [34, 35] and the formal specification of the heart to model a closed-loop system for verifying the desired behaviour of the cardiac pacemaker for certification purpose.

References

1. Jaakko Malmivuo, R.P.: Bioelectromagnetism. Oxford University Press (1995) ISBN 0-19-505823-2
2. Khan, M.G.: Rapid ECG Interpretation. Humana Press (2008)
3. Maisel, W.H., Sweeney, M.O., Stevenson, W.G., Ellison, K.E., Epstein, L.M.: Recalls and safety alerts involving pacemakers and implantable cardioverter-defibrillator generators. JAMA: The Journal of the American Medical Association 286(7), 793–799 (2001)
4. Center for Devices and Radiological Health: Safety of Marketed Med. Devices, FDA (2006)
5. A Research and Development Needs Report by NITRD: High-Confidence Medical Devices: Cyber-Physical Systems for 21st Century Health Care, `http://www.nitrd.gov/About/MedDevice-FINAL1-web.pdf`
6. Keatley, K.L.: A review of the fda draft guidance document for software validation: guidance for industry. Qual. Assur. 7(1), 49–55 (1999)
7. Lee, I., Pappas, G.J., Cleaveland, R., Hatcliff, J., Krogh, B.H., Lee, P., Rubin, H., Sha, L.: High-confidence medical device software and systems. Computer 39(4), 33–38 (2006)
8. Méry, D., Singh, N.K.: Trustable Formal Specification for Software Certification. In: Margaria, T., Steffen, B. (eds.) ISoLA 2010. LNCS, vol. 6416, pp. 312–326. Springer, Heidelberg (2010)
9. Bowen, J., Stavridou, V.: Safety-critical systems, formal methods and standards. Software Engineering Journal 8(4), 189–209 (1993)
10. Jetley, R.P., Carlos, C., Iyer, S.P.: A case study on applying formal methods to medical devices: computer-aided resuscitation algorithm. STTT 5(4), 320–330 (2004)
11. Jetley, R., Purushothaman Iyer, S., Jones, P.: A formal methods approach to medical device review. Computer 39(4), 61–67 (2006)
12. Méry, D., Singh, N.K.: Real-time animation for formal specification. In: Aiguier, M., Bretaudeau, F., Krob, D. (eds.) Complex Systems Design & Management, pp. 49–60. Springer, Heidelberg (2010)
13. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press (2010)
14. Fitzgerald, J.: Logics of Specification Languages. In: Bjørner, D., Henson, M.C. (eds.) The Typed Logic of Partial Functions and the Vienna Development Method. EATCS Textbook in Computer Science, pp. 431–465. Springer, Heidelberg (2007)
15. Harrild, D.M., Henriquez, C.S., Atria, T.H., Harrild, D.M., Henriquez, C.S.: Cs, a computer model of normal conduction. The Human Atria, Circ. Res. 87, 25–36 (2000)
16. Bayes de Luna, A., Batcharov, V.N., Malik, M.: The morphology of the Electrocardiogram. In: The ESC Textbook of Cardiovascular Medicine. Blackwell Publishing Ltd. (2006)
17. von Neumann, J.: Theory of Self-Reproducing Automata. University of Illinois Press (1966); edited by Burks, A.W.
18. Artigou, J.-Y., Monsuez, J.-J., Societe française cardiologie: Cardiologie et maladies vasculaires. Elsevier Masson (2006)
19. Project RODIN: Rigorous open development environment for complex systems (2004), `http://rodin-b-sharp.sourceforge.net/`

20. Plonsey, R., Barr, R.C.: Mathematical modeling of electrical activity of the heart. Journal of Electrocardiology 20(3), 219–226 (1987)
21. Kye-Rok Jun, Y.R.S., Kim, T.G.: A cellular automata model of activation process in ventricular muscle. In: SCSC 1994, pp. 769–774 (1994)
22. Berenfeld, O., Abboud, S.: Simulation of cardiac activity and the ecg using a heart model with a reaction-diffusion action potential. Medical Engg. & Physics 18(8), 615–625 (1996)
23. Adam, D.: Propagation of depolarization and repolarization processes in the myocardium-an anisotropic model. IEEE Transactions on Biomedical Engg. 38(2), 133–141 (1991)
24. Jiang, Z., Pajic, M., Connolly, A.T., Dixit, S., Mangharam, R.: Real-time heart model for implantable cardiac device validation and verification. In: 22nd Euromicro Conference on Real-Time Systems (IEEE ECRTS 2010) (July 2010)
25. Barold, S.S., Stroobandt, R.X., Sinnaeve, A.F.: Cardiac Pacemakers Step by Step. Futura Publishing (2004) ISBN 1-4051-1647-1
26. Ellenbogen, K.A., Wood, M.A.: Cardiac Pacing and ICDs, 4th edn. Blackwell (2005) ISBN-10 1-4051-0447-3
27. Hesselson, A.: Simplified Interpretations of Pacemaker ECGs. Blackwell Publishers (2003) ISBN 978-1-4051-0372-5
28. Lee, I., Pappas, G.J., Cleaveland, R., Hatcliff, J., Krogh, B.H., Lee, P., Rubin, H., Sha, L.: High-confidence medical device software and systems. Computer 39(4), 33–38 (2006)
29. Love, C.J.: Cardiac Pacemakers and Defibrillators. Landes Bioscience Publishers (2006) ISBN 1-57059-691-3
30. Makowiec, D.: The Heart Pacemaker by Cellular Automata on Complex Networks. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) ACRI 2008. LNCS, vol. 5191, pp. 291–298. Springer, Heidelberg (2008)
31. Back, R., von Wright, J.: Refinement Calculus A Systematic Introduction. Graduate Texts in Computer Science. Springer (1998)
32. Méry, D., Singh, N.K.: Technical Report on Formalisation of the Heart using Analysis of Conduction Time and Velocity of the Electrocardiography and Cellular-Automata. Technical report (2011), http://hal.inria.fr/inria-00600339/en/
33. Clarke, E.M., Grumberg, O., Peled, D.: Model Checking. MIT Press (1999)
34. Méry, D., Singh, N.K.: Functional behavior of a cardiac pacing system. International Journal of Discrete Event Control Systems 1(2), 129–149 (2011)
35. EB2ALL: Automatic code generation from Event-B to many Programming Languages (2011), http://eb2all.loria.fr/

An Event-B Approach to Timing Issues Applied to the Generic Insulin Infusion Pump

Hao Xu and Tom Maibaum

Department of Computing and Software
McMaster University
xuh32@mcmaster.ca, tom@maibaum.org

Abstract. An insulin infusion pump (IIP) is a complicated and time critical control system. Making sure that the pump infuses insulin in conformance with a user's wishes and in conformance with safety related constraints, and does so at the right times, makes it a highly safety critical system. This paper uses Event-B to specify a generic model for an IIP, based on requirements developed by the US Food and Drug Administration (FDA). The IIP is an active and reactive control system. Each transition between states of the model is modelled as an event. To correctly specify the IIP, we need a model of time and synchronization of events with time that is sufficiently rich to achieve our safety aims. We create several sets to model the activation times of different events and the union of these time sets defines a global time activation set. All the actions in an event are triggered only when the global time matches the time specified in the event. When the action is activated, the time is deleted from the corresponding time set, but not the corresponding global time set. A time point is deleted from the global time set only when there are no pending actions for that time point. We are able to demonstrate that the resulting specification satisfies relevant required safety constraints.

Keywords: insulin infusion pump, Event-B, safety critical systems, safety constraints, formal specification, timing constraints.

1 Introduction

1.1 Motivation

The Insulin infusion pump (IIP) is a semi-automatic, patient controlled medical device whose purpose is to control the blood glucose (BG) level of diabetics by continuously or intermittently infusing insulin from an insulin reservoir into the patient. An IIP simulates the behaviour of the pancreas, which accurately delivers (using a biological feedback mechanism) insulin to the body during normal activities of human beings. The accuracy and the complexity of the (healthy) pancreas in regulating biological behaviour and the essential role it plays makes the use of IIP, as an artificial replacement, a high risk system. A recent report from the FDA indicates that over 5000 adverse events for Insulin

Pumps were reported during 2008 [1]. The effects of incorrect infusion (too much or too little or at the wrong time) may be catastrophic for the patient. The faults and errors that occur can result either from system errors, i.e., incorrect or unsafe device behaviour, or from inappropriate behaviour on the part of users, i.e., mistakes in programming the device or malicious use by the user or others. To study how to avoid the occurrence of faults and errors in IIP systems, an IIP project was started in the McMaster Centre for Software Certification (McSCert). Our objective is to build models of an IIP that reduce the occurrence of faults and errors assignable to the system and to make it possible to demonstrate to regulators, such as the FDA, that an IIP device is safe to license. (We are not in this paper going to address user interface issues or potential purposeful misuse by the patient or others.) To judge the correctness of a model is far easier than making the same judgment directly about the program [2], so our strategy for implementing IIP systems is to create a generic and abstract model for it. An IIP is a time critical control system; clarifying the timing properties and proposing a suitable timing model is one of the significant features of any model of an IIP. The generic model, which this paper presents, mainly focuses on the timing issues of an IIP. The full version of the IIP model can be found in [3].

1.2 Overview of Event-B

Event-B is an event based extension of the B method [4], which was developed by Jean-Raymond Abrial some years ago [2]. Its mathematical notations are based on propositional logic and Zermelo-Fraenkel set theory [5].

Each Event-B model is constructed in terms of *states* and *events*. The states of the model include a static part (*constants, carrier sets*) and a dynamic part (*variables*). The events are interpreted by guards (trigger conditions of actions) and actions (transitions between states). The notation for an Event-B model has two components: *context* and *machine*. The context is made up of a static part and the relevant properties of the static part, namely *constants, carrier sets, axioms* and *theorems*. A context can *extend* one or several contexts. There may be no context in a model. The machine describes the dynamic part, focusing on events and the internal properties of the system, namely *variables*, *invariants*, *variants* and *events*. Machines can be related by refinement. A machine can *refine* only one other machine. The relation between a machine and its context is named *sees*; a machine can see several contexts or no context. The proof obligation rules for Event-B are used to automatically generate properties that must be proved about Event-B specifications (proof obligations (POs)), with the purpose of checking the consistency of the system and the invariant preservation properties for the model. Subsequently, mathematical proofs must be created for these POs.

Event-B is a tool supported formal specification language. The Rodin platform [6], an Eclipse-based IDE for Event-B, provides a user friendly user interface to create, refine and mathematically prove properties of models. As noted above, proof obligations are automatically generated while building models. The most powerful aspect of the Rodin platform is that a semi-automatic theorem prover is built into it, which saves the user lots of time by automatically generating a large majority of the proofs required.

1.3 Summary

This paper proposes a generic Event-B approach to handle the timing issues applied to the IIP. To handle the timing issues we create a timing pattern and some variations of it by exploiting Event-B notation in section 2. These patterns are not only suitable for the IIP model, but also adapt to other time critical systems. A brief description (requirements related to timing) of the IIP is given in section 3. The 4th section describes the generic model for the IIP, which is a practical application of the timing approach to the IIP. Each part of the model in section 4 can be traced to one or several corresponding requirements in section 3, which demonstrates that our specification satisfies the requirements. The POs associated with this model have all been proved, which indicates the model is internally consistent.

2 A Time Pattern for Event-B

As we mentioned above, Event-B has a number of expressive features. To use Event-B for systems where timing is important, it is necessary to find ways of expressing time related properties in Event-B specifications. Dominique Méry and Joris Rehm in their papers [7,8] described a solution for specifying time constraints in Event-B. The proposed solution is to create a time activation set, which records all the activation time points of the system. Activation time points are represented by natural numbers, which indicates that the model is a discrete time model. There are two kinds of events associated with the time pattern. One kind creates the activation time set; the second kind is used to delete a time point from the activation time set. Each event related to a predictable time point is guarded by some time constraint. Once the specified time constraint is satisfied for some event, the relevant action is triggered and the time point is deleted from the activation time set. Time elapsing is specified in terms of a special event *tick_tock*. For any time that is smaller than the minimum value in the time set, the current time will jump to that value in one step without taking on any intervening time value. However, there exist some shortcomings in this pattern. The pattern cannot handle those events whose trigger time is unpredictable. Nor does it handle the problem of multiple events using the same time point to trigger a transition.

Prompted by the shortcomings of the pattern described above, we introduce two terms and their definitions. *Predictable events* are those events whose occurrence and timing can be derived from the user defined program. *Unpredictable events* are those events whose occurrence time is unpredictable. In other words, unpredictable events are those events that we do not know whether or when they occur. Examples of such unpredictable events includes user initiated events such as powering off, pausing the IIP, etc. In this paper we make an improvement to the pattern in [7] by adding mechanisms for dealing with unpredictable events and by modifying the specification of the *tick_tock* event to a self incrementing clock. The new time pattern uses discretized time, as with the previous pattern, to approximately represent continuous time. The precision of the minimum system time increment is given by some fixed quantum in terms of each *tick_tock* event, so time increasing in the model is approximately smooth. Here we use *smooth* to represent the phrase: time increasing by one unit increment. Smoothly increasing the time is a sufficiently good way to handle and capture the unpredictable discrete events in the model.

2.1 A Time Pattern with Unpredictable Events

The core drivers for this pattern are the unpredictable event and the new *TICK_TOCK* event. Because a model may contain lots of unpredictable events, we present a general schema for the time pattern.

```
MACHINE  Time Pattern
VARIABLES
    time
    at
    v
INVARIANTS
    inv1 : $time \in \mathbb{N}$
    inv2 : $at \subseteq \mathbb{N}$
    inv3 : $inv(v_i)$
    inv4 : $at \neq \varnothing \Rightarrow$
        $time \leq min(at) \vee \neg p(v_i)$
EVENTS
Initialisation
    begin
        act1 : $time := 0$
        act2 : $at := \varnothing$
        act3 : $v_i : | v_i' \cdot p(v_i)$
    end
Event  POST_TIME $\widehat{=}$
    any
        tm
    where
        grd1 : $tm \in \mathbb{N}$
        grd2 : $tm > time$
    then
        act1 : $at := at \cup \{tm\}$
    end
Event  UNPRED_EVT i $\widehat{=}$
    when
        grd1 : $p(v_i)$
    then
        act1 : $v_i : | v_i' \cdot \neg p(v_i)$
    end
Event  UNPRED_EVT j $\widehat{=}$
    when
        grd1 : $\neg p(v_i)$
        grd2 : $at \neq \varnothing$
    then
        act1 : $v_i : | v_i' \cdot p(v_i)$
        act2 : $at : | at' =$
            $\{x | x \in at \wedge x > time\}$
    end
Event  PROCESS_TIME $\widehat{=}$
    when
        grd1 : $time \in at$
        grd2 : $p(v_i)$
    then
        act1 : $at := at \setminus \{time\}$
    end
Event  TICK_TOCK $\widehat{=}$
    when
        grd1 : $at \neq \varnothing \Rightarrow$
            $time < min(at) \vee \neg p(v_i)$
    then
        act1 : $time := time + 1$
    end
END
```

Note that the words in uppercase, such as *VARIABLES, INVARIANTS, EVENTS, etc* are the keywords of the Event-B notation. The keywords in lowercase, such as *grd, act, any* are used to indicate the *guards, actions* and *abstract parameters* in Event-B. The following symbols are special symbols used in this pattern, other than the keywords of the Event-B model notation.

Definition of Symbols

$time$ – current time
at – activation time set
v – a list of variables v_1, v_2, v_3, v_i...
$inv(v_i)$ – the corresponding invariants of the element v_i in v
$p(v_i)$ – predicate of a variable v_i in v, which is used as a guard of some predictable events (and may disable the predictable events)

Description of the Pattern

This pattern deals with both predictable events and unpredictable events.

POST_TIME are those events which create or change at: a time point which is bigger than the current time shall be added to at. This event creates the activation time set for the predictable events, so the unpredictable time points are not included in at.

PROCESS_TIME describes the predictable events; it simply specifies the state transitions of the model when the current time corresponds to a predictable time point in at.

The *UNPRED_EVT* indicates an event which is unpredictable. We put i, j... after *UNPRED_EVT* to show there may exist several unpredictable events in the model. These unpredictable events can appear independently or in pairs. If *UNPRED_EVT_i* appears independently, in this pattern, events associated with *PROCESS_TIME* will never be triggered because of the guard $p(v_i)$. If these events appear in pairs, then *UNPRED_EVT_j* sets v_i back to the value which satisfies $p(v_i)$. At the same time, the time points in at that are smaller than the current time shall be deleted from at because the relevant time has already elapsed.

The expression $v_i :| v_i' \cdot p(v_i)$ is the notation called before-after predicate assignment in Event-B, meaning v_i is assigned a value that satisfies $p(v_i)$.

The $inv4$ says that if at is not empty, then the current time is smaller than or equal to the minimum value of at or $p(v_i)$ is not true. The reason we add "$\vee\neg p(v_i)$" in $inv4$ is that *TICK_TOCK* may be disabled when $time = min(at)$ under the circumstance that the predictable events are disabled by some unpredictable events. Therefore, except when $p(v_i)$ is true, $time$ must be at most the minimum of at.

Note that the pattern allows some variations. For example, some of the events can be merged into a single event, such as merging *POST_TIME* and *UNPRED_EVT* into one event.

2.2 Time Pattern for Classifying the Events

This section discusses another issue often occurring in a time related system – two, or even more events, triggering simultaneously (relative to the notion of time being used in the model).

If we use the time pattern discussed in [7,8] to specify this kind of issue, some problems may occur. These problems are a consequence of the state transition mechanism of Event-B – only one enabled event will be randomly picked and executed for each state transition in the model. Suppose we have two events *PROCESS_TIME_A* and *PROCESS_TIME_B*. One of the guards in these events is $time = time' \wedge time' \in at$. When this time constraint is satisfied, these two events are enabled and will be queued for execution. Event-B will randomly pick one of them, say *PROCESS_TIME_A*, and execute its actions. But examining the pattern, we see one of the actions is $at := at \setminus \{time\}$. Because Event-B does not support multi-sets, $time'$ will no longer be included in at. The triggering of event *PROCESS_TIME_A* directly results in another event *PROCESS_TIME_B* being disabled. Because the time variable *time* continually increases in *TICK_TOCK*, the *PROCESS_TIME_B*, which we are still expecting to be executed, will never happen.

In trying to solve this kind of problem in Event-B, we have come up with two solutions.

Solution I: Combining the Events. If we combine these events, whose guards include the same time constraints, together into one event, the time pattern discussed in section 2.1 is an effective way of addressing this issue. Note that combining events is dangerous, because it does not mean that the old events shall disappear. The old events may still be kept in the model, with a slight change to the guard.

For example, if we have two different events A and B that both have time constraint $time \in a$ and $time \in b$ (a, b both correspond to sets of time points), receptively, and $a \cap b = c$, then the guards of A, B include the same time constraint: $time \in c$. To combine A, B, we create a new event *A_B* with time constraint $time \in c$. At the same time, the guards of old events A and B should be changed to $time \in a \setminus c$ and $time \in b \setminus c$ respectively. Because c may be equal to a or b, a new issue of deciding whether to keep the old event or not arises.

Moreover, in practice, especially in the development of large scale systems, it is really difficult to analyze all the possibilities involving events with overlapping time constraints, never mind the problem of combining them together. Even if the developer successfully combines these events into one, checking the correctness of this event and those modified relevant old events is an enormous task. This solution potentially increases the safety risk and the complexity of the system, so we are not recommending it here.

Solution II: Classifying Events. This solution is a modification of the time pattern with unpredictable events. For a large model including a large number of events, we can classify the events into several categories. The criterion for the classification is derived from the variables. For example, suppose we have *PROCESS_TIME_A* and *PROCESS_TIME_B* in the model. We can find the corresponding events such as *POST_TIME_A* and *POST_TIME_B* in the model. Instead of using *at* as the activation time set we define, for each category, an independent time set. The union of these sets defines the global time set *at*. Consequently, we create independent sets for the *POST_TIME_A/B* events and delete the time point from each of the corresponding time activation sets in the *PROCESS_TIME_A/B* event, as appropriate instead of deleting the time point from the global time activation set. Only when all of the events in the model sharing a given time constraint are executed, will the *time* in *TICK_TOCK* increase and the corresponding time will then be deleted from *at*. Therefore, the most important invariant in the time pattern will also be modified to:

$$at\_a \cup at\_b \cup ... \neq \varnothing \Rightarrow time \leq min(at\_a \cup at\_b \cup ...) \vee \neg p(vi)$$

We present below the modification of those events which are used to handle this time issue.

```
Event  POST_TIME_A ≙
    any
        tm1
    where
        grd1 : $tm1 \in \mathbb{N}$
        grd2 : $tm > time$
    then
        act1 : $at\_a := at\_a \cup \{tm1\}$
    end
Event  POST_TIME_B ≙
    any
        tm2
    where
        grd1 : $tm2 \in \mathbb{N}$
        grd2 : $tm2 > time$
    then
        act1 : $at\_b := at\_b \cup \{tm2\}$
    end
Event  PROCESS_A ≙
    when
        grd1 : $time \in at\_a$
    then
        act1 : $at\_a := at\_a \setminus \{time\}$
    end
Event  PROCESS_B ≙
    when
        grd1 : $time \in at\_b$
    then
        act1 : $at\_b := at\_b \setminus \{time\}$
    end
Event  TICK_TOCK ≙
    when
        grd1 : $at\_a \cup at\_b \cup ... \neq \varnothing \Rightarrow$
          $time < min(at\_a \cup at\_b \cup ...)$
          $\vee \neg p(vi)$
    then
        act1 : $time := time + 1$
    end
```

In *POST_TIME_A*, we change $at := at \cup \{tm\}$ to $at\_a := at\_a \cup \{tm1\}$. Correspondingly, *PROCESS_TIME_A* deletes the current time from *at_a* (the independent time activation set for A). We make a similar changes to *POST_TIME_B* and *PROCESS_TIME_B*. The guard of the *TICK_TOCK* event uses $at\_a \cup at\_b \cup$ ... instead of *at*. Note that we are not using $p(v_i)$ in *PROCESS_TIME_A* and *PROCESS_TIME_B* but keeping it instead in *TICK_TOCK*, because $p(v_i)$ has no relation with this time issue. Its occurrence or non occurrence has no effect on the pattern. On the other hand, *TICK_TOCK* describes the behaviour of *time*; it will be constrained by all the other $p(v_i)$ which may appear in events that are similar to *PROCESS_TIME*.

2.3 Comparison with Classic Timed Automata

This section makes a comparison of our solution with classic timed automata [9,10].

Classic timed automata use continuous time in the form of the real numbers ($\mathbb{R}$) to describe state transitions. More than one *clock* can be specified in the automaton. The time constraints on the transitions and states restrict the system transit time. A system transition happens when the passage of time satisfies the time constraint on the transition, otherwise the system will stay in the previous state. Moreover, the time on the transitions can be reset to zero in classic timed automata.

In our solution, time is described in a discretized way ($\mathbb{N}$). The time concept in our solution is quite general. The elements in the global time set indicate the predictable system transition time points. Time elapsing is specified by an event *tick_tock*, so we only have one clock in our solution. The system transitions happen when the time reaches a predictable time point. The constraints on transitions in classic timed automata are interpreted by the guards in our predictable events. Those unpredictable events with unpredictable trigger time are constrained by environmental actions other than the elements in the activation time set. Because time elapsing is interpreted by a general self increment clock in our pattern, we never reset the time variable to zero.

3 Description of an IIP

An IIP has a large number of features, and even a generic IIP has lots of complicated requirements or constraints. Because the focus of this paper is the timing related issues, we only describe the time related requirements of IIP. The following requirements are extracted from [1].

Our model of IIP focuses on the controller of the IIP. The controller takes information from the environment (here we are not considering sensors, so only the user actions have an effect on the controller) as input and changes the states of corresponding variables in conformance with the timing requirements as output. The performance of the IIP is based on internal algorithms for handling flow rate and functionalities for handling user behaviour. These are the relevant predictable events and unpredictable events referred to in later sections. The following are some general requirements of IIP:

- The pump can be turned on and off, be paused and be resumed. (*IIP-req-1*)
- When the pump is turned on, the flow rate shall be set to the basal rate defined in the basal profile, in accordance with the times determined by the profile (a combination of times defined by the user and the times calculated by the alogrithms internal to the IIP). (*IIP-req-2*)
- Basal rate changes shall satisfy the requirements of the basal profile (as different amounts may be required at different times of the day). (*IIP-req-3*)
- The combined flow rate shall be equal to the basal rate required plus the extended bolus rate defined by the user, except in the case when the user requests a normal bolus. (*IIP-req-4*)
- When the pump is paused, all the requested actions shall be forbidden and the combined flow rate shall be set to zero. (*IIP-req-5*)
- When the pump is paused and the extended bolus is in progress, the extended bolus infusion shall be stopped and the amount of extended bolus that has been delivered shall be shown to the user. (*IIP-req-6*)
- When the pump is resumed, the flow rate shall be set to the basal rate in the basal profile, but adjusted to the current time (i.e., the basal insulin that

would have been delivered between the time the pump is paused and the time it is resumed is no longer to be delivered). (*IIP-req-7*)
– When the pump is off, all user defined profiles are cleared and the combined flow rate set to zero. (*IIP-req-8*)

The main behaviours of the IIP are described in terms of two kinds of insulin infusion processes, called *basal* and *bolus*. We present the detailed requirements of *basal* and *bolus* behaviours next.

The basal insulin delivery, which supports the base insulin requirements in the daily life of a user, is of long duration, with small amounts of insulin being infused into the user over the whole day. Because the amount of basal insulin requested fluctuates over periods in a day, a suitable basal profile will be set by the user. The following are the requirements for a basal profile:

– The basal profile shall cover 24 hours. (*bp-req-1*)[1]
– Each basal segment shall have a start time, a duration time and a basal rate. (*bp-req-2*)
– There is no overlap or gap between adjacent basal segments. (*bp-req-3*)

The bolus insulin delivery, which is used to deal with food intake, relative to estimates of exercise levels, has a short infusion period, and has relatively large amounts of insulin being processed. The bolus process includes two variants: an instant insulin delivery (normal bolus) and an extended insulin delivery with duration (extended bolus). The requirements for bolus infusions are:

– The bolus infusion includes normal bolus and extended bolus. (*bo*[2]*-req-1.*)
– The user shall specify the total amount of bolus. (*bo-req-2*)
– The user shall specify the amount of bolus for normal bolus. (*nbo*[3]*-req-1.*)
– The normal bolus is an instant injection of a specified amount of insulin. (*nbo-req-2*)
– The extended amount of bolus is equal to the total amount of bolus minus the amount of normal bolus. (*exbo*[4]*-req-1.*)
– The user shall specify the start time and the duration for extended bolus if the extended bolus amount is not equal to 0. (*exbo-req-2*)
– The extended bolus shall be delivered evenly during the duration the user specified. (*exbo-req-3*)

The requirements above are a brief description of the IIP. We only consider the properties related to timing issues; requirements such as the combined flow rate being limited by a maximum flow rate are not considered here. From the description of the system, we can classify the events of the system into two categories: insulin processing and pump processing. The insulin processing can also be classified into basal, normal bolus and extended bolus processing.

[1] bp-req-1 stands for the basal profile requirement 1.
[2] bo stands for bolus.
[3] nbo stands for normal bolus.
[4] exbo stands for extended bolus.

4 Modelling an IIP in Event-B

The requirements describe simple functionalities of the IIP. Before we construct the model, we reiterate that this paper focuses only on the timing issues and the main functionalities of the IIP. Because of space limitations, we ignored many features such as power on and off, power on self test check, the pump priming process, etc. This model only describes the simplest behaviour of the controller that relates to timing issues. Because this model is a component of the whole system, some quantities used in the *context* would be moved to the *machine* in a full specification. The full specification of the IIP [3] is far more complicated than the one we present below.

4.1 Formalizing the States

In this section we formally describe the *context*, the *variables* and the *invariants* of the IIP model. Before formalizing the state, we shall clarify which parameters are static and which are dynamic.

From the description of the last section, we see that, although the basal profile is set by the user, insulin processing behaviors all assume that the basal profile has already been set. Therefore, the basal profile shall be the static part of our (limited) model.

As can been seen below, the *IIP context* indicates the name of the context. We introduce an enumeration type *state*, which has two constants: *on* and *off* (axm4, 5). The constant set *state* can be used to represent the states of a number of variables, such as the pump state or pause state.

CONTEXT IIP Context
SETS
    state
CONSTANTS
    ba_t
    ba_rs
    bpf
    on
    off

AXIOMS
    axm1 : $ba\_t \subseteq \mathbb{N} \wedge 0 \in ba\_t$
    axm2 : $ba\_rs \subseteq \mathbb{N}_1 \wedge ba\_rs \neq \varnothing$
    axm3 : $bpf \in ba\_t \rightarrow ba\_rs$
    axm4 : $state = \{on, off\}$
    axm5 : $on \neq off$
END

The interesting issue here is the attributes being used to define the basal profile. Event-B is an event based specification language, which is only concerned with the discrete events in the model. By assessing *bp-req-1,2,3* in light of the event based features of Event-B, it may be inferred that it is unnecessary to know the duration for each basal segment. Just the start time and the corresponding basal rate are enough to describe the basal profile. Therefore, we define the basal profile (bpf) as a total function from the start time set (ba_t) for each basal segment to the basal rate set (ba_rs) (axm3). The total function indicates the start time is unique, but the relative basal rate is not. For simplicity, we define the start time set (ba_t) of the basal profile as a subset of the natural numbers (axm1) instead of considering the unit of time and the translation between days, hours, etc. Note that the reason why we include 0 in *ba_t* is that, when the pump is turned on there shall be a corresponding value for basal at that time (*IIP-req-1*), so that the *ba_t* shall begin with 0. The axm2 defines *ba_rs* as a subset of the positive natural numbers and that it is not an empty set.

The dynamic states of the model are interpreted by variables. Most of the variables in the model correspond to the quantities mentioned in the requirements. Some of the auxiliary variables, such as pump state (*p_stat*) and pause state (*paus_stat*), are introduced into the model with the intention of making the specification clear. Current time (t) is a variable added into the model to enable us to handle timing issues, which is not mentioned in the requirements. The variables and invariants are shown below:

VARIABLES

```
    rate //combined flow rate
    ba_r //basal rate
    ba_at //basal time set
    p_stat //pump state
    nbo_am //bolus amount
    exbo_r //ext bolus rate
    exbo_a //ext bolus start time
    exbo_t //ext bolus stop time
    exbo_at //ext bolus time set
    exbo_req //ext bolus request
    exbo_am //ext bolus amount
    paus_stat //pause state
    t //current time
```

INVARIANTS

inv1 : $rate \in \mathbb{N}$

inv2 : $ba\_r \in ba\_rs \vee ba\_r = 0$

inv3 : $ba\_at \subseteq \mathbb{N}$

inv4 : $p\_stat \in state$

inv5 : $nbo\_am \in \mathbb{N}$

inv6 : $exbo\_r \in \mathbb{N}$

inv7 : $exbo\_a \in \mathbb{N}$

inv8 : $exbo\_t \in \mathbb{N}$

inv9 : $exbo\_a \leq exbo\_t$

inv10 : $exbo\_at \subseteq \mathbb{N}$

inv11 : $exbo\_req \in \mathbb{N}$

inv12 : $exbo\_am \in 0 \,..\, exbo\_req$

inv13 : $paus\_stat \in state$

inv14 : $rate = ba\_r + exbo\_r$

inv15 : $t \in \mathbb{N}$

inv16 : $ba\_at \cup exbo\_at \neq \varnothing \Rightarrow t \leq min(ba\_at \cup exbo\_at) \vee paus\_stat = on$

The intended meanings of the variables are indicated in the comments beside them. From the variable list we see some of the variables just simply change their format, without modifying the intended meaning in the requirements. For instance, although the extended bolus duration (*exbo-req-2*) does not appear in the variable list, *exbo_a*, *exbo_t* are declared and the *inv9* indicates the internal relation between them. The inclusion of the variables *nob_am* and *exbo_req* in the model captures the requirement *bo-req-1*. The variable *exbo_r* indicates that the extended bolus shall be delivered evenly at a certain extended bolus rate (*exbo-req-3*).

The corresponding invariants of the variables indicate that some requirements are the invariants of the model, for instance *inv14* matches *IIP-req-4*. For some of the variables, whose interpretations are quite straightforward, we are not explaining their role here. The *inv12* is an internal property for extended bolus. The *exbo_am* is the extended bolus amount the user gets while the *exbo_req* is the extended bolus amount the user requested. The received extended bolus amount shall be smaller than or equal to the required extended bolus amount. The *inv16* is the most interesting invariant. By comparing it with the time pattern we mentioned in section 2, we see that this model not only includes unpredictable events, but that we also need to classify events to properly handle the timing issue. The predicate $p(v_i)$ used in the timing pattern for an unpredictable event that may result in the *TICK_TOCK* event being disabled in this model is $paus\_stat = on$. From *inv16*, we see that the events in this model shall be classified into two groups, namely those events relevant to basal and those events relevant to extended bolus. Two time activation sets are created to handle the issue of events triggering at the same time in this model.

4.2 Formalizing the Events

The dynamic states and their properties (invariants) combined with events comprise the Machine (*IIP Machine*) of this model. In this model, the *IIP Machine* *sees* (keyword in Event-B) *IIP Context*, which means the *IIP Machine* can use the static quantities in the *IIP Context*.

For each Event-B machine, an event called *INITIALISATION* is included, giving the initial state of the machine.

```
MACHINE IIP Machine
SEES IIP Context
EVENTS
Initialisation
    begin
        act1 : $rate := 0$
        act2 : $ba\_r := 0$
        act3 : $ba\_at := \varnothing$
        act4 : $p\_stat := off$
        act5 : $nbo\_am := 0$
        act6 : $exbo\_r := 0$
        act7 : $exbo\_a := 0$
        act8 : $paus\_stat := off$
        act9 : $exbo\_t := 0$
        act10 : $exbo\_at := \varnothing$
        act11 : $exbo\_req := 0$
        act12 : $exbo\_am := 0$
        act13 : $t := 0$
    end
```

The initialization event in this model is quite straightforward, just assigning all natural number variables the value 0, all set variables are assigned ∅ and the state variables are set to *off*.

From *IIP-req-1*, we need an *ON* event here to indicate the state transition when the pump starts to infuse. Event-B uses the key word *any* to indicate the *abstract parameter*. We always use the abstract parameter to describe the value obtained from the environment. Here *ct* stands for the current time value input from outside. If *ct* is a natural number, then current time *t* is set as *ct*.

```
Event ON $\widehat{=}$
    any
        $ct$
    where
        grd1 : $ct \in \mathbb{N}$
        grd2 : $ba\_at = \varnothing$
        grd3 : $exbo\_at = \varnothing$
        grd4 : $p\_stat = off$
        grd5 : $paus\_stat = off$
        grd6 : $exbo\_r = 0$
    then
        act1 : $t := ct$
        act2 : $rate := bpf(max(\{i|i \in ba\_t \land i \leq ct\}))$
        act3 : $ba\_at := ba\_t \setminus \{i|i \in ba\_t \land i \leq ct\}$
        act4 : $ba\_r := bpf(max(\{i|i \in ba\_t \land i \leq ct\}))$
        act5 : $p\_stat := on$
    end
```

To capture *IIP-req-2*, we set both the basal rate and combined flow rate to the value corresponding to the biggest time point that is smaller than or equal to the current time in *bpf*. Comparing this event to the time pattern, we see the strategy of classifying time activation sets through variables is used for basal. The "\" symbol is the set subtract symbol. The basal time set *ba_at* is assigned a set which contains the elements in *ba_t* by subtracting the elements smaller than or equal to the current time.

```
Event  $BA\_CHANGE \mathrel{\widehat{=}}$
    when
        grd1 : $t \in ba\_at$
        grd2 : $rate \geq ba\_r$
        grd3 : $ba\_at \subseteq ba\_t$
        grd4 : $paus\_stat = off$
    then
        act1 : $rate := rate - ba\_r + bpf(t)$
        act2 : $ba\_r := bpf(t)$
        act3 : $ba\_at := ba\_at \setminus \{t\}$
end
```

The *BA_CHANGE* event is a typical *PROCESS_TIME* event mentioned in the time pattern. It is constrained by the guard $paus\_stat = off$. Once the pump is paused, this event will be disabled. *BA_CHANGE* is also a representation of *IIP-req-3*. When the current time increases to a value that is equal to an element in *ba_at*, then *ba_r* changes to $bpf(t)$. In other words, the previous basal rate value is substituted by a new basal rate (*bpf(t)*) corresponding to the current time. Therefore, the combined flow rate (*rate*) is changed to "*rate-ba_r+bpf(t)*". The time *t* is deleted from *ba_at*.

```
Event  $NBO\_DEL \mathrel{\widehat{=}}$
    any
        $BO\_AM$
        $NBO\_AM$
        $EXBO\_A$
        $EXBO\_D$
    where
        grd1 : $BO\_AM \in \mathbb{N}_1$
        grd2 : $NBO\_AM \in \mathbb{N}_1 \land NBO\_AM \leq BO\_AM$
        grd3 : $EXBO\_A \geq t$
        grd4 : $EXBO\_D \in \mathbb{N}_1$
        grd5 : $exbo\_at = \varnothing$
        grd6 : $ba\_at \cup exbo\_at \neq \varnothing \Rightarrow t \leq min(ba\_at \cup exbo\_at)$
        grd7 : $paus\_stat = off$
    then
        act1 : $nbo\_am := NBO\_AM$
        act2 : $exbo\_req := BO\_AM - NBO\_AM$
        act3 : $exbo\_a := EXBO\_A$
        act4 : $exbo\_t := EXBO\_A + EXBO\_D$
        act5 : $exbo\_at := \{EXBO\_A, EXBO\_A + EXBO\_D\}$
        act6 : $exbo\_am := 0$
end
```

Firstly, *NBO_ON* satisfies *bo-req-1,bo-req-2,nbo-req-1*: the total amount of bolus is indicated by *BO_AM* and the amount of normal bolus is represented by *NBO_AM*. It also satisfies *nbo-req-2* and *exbo-req-1,2*. The normal bolus is delivered instantly, upon getting the amount of normal bolus from the environment (act1). The required extended bolus is calculated from the bolus amount minus the normal bolus amount (act2). This event also creates the activation time set for extended bolus (act5). Because the setting of extended bolus is performed by the user and can only be executed once or never (when the amount of normal bolus is equal to the total bolus amount), after each normal bolus infusion, the *exbo_at* only has two elements, namely start time (*exbo_a*) and stop time (*exbo_t*). From this, we can see why we specify independent time activation sets for both basal and extended bolus. Because *ba_at* and *exbo_at* may have a non empty intersection, the *BA_CHANGE* and *EXBO_ON* or *EXBO_OFF* may have the same time constraint. Creating the category of time activation sets prevents any of above events from being disabled.

```
Event EXBO_ON ≙
  when
    grd1 : $t = exbo\_a \wedge t \in exbo\_at$
    grd2 : $exbo\_at \neq \varnothing \wedge$
           $exbo\_t > exbo\_a$
    grd3 : $paus\_stat = off$
    grd4 : $exbo\_r = 0$
  then
    act1 : $exbo\_r :=$
        $exbo\_req/(exbo\_t - exbo\_a)$
    act2 : $rate := rate +$
        $exbo\_req/(exbo\_t - exbo\_a)$
    act3 : $exbo\_at :=$
        $exbo\_at \setminus \{exbo\_a\}$
    act4 : $exbo\_am := 0$
  end
Event EXBO_OFF ≙
  when
    grd1 : $t = exbo\_t \wedge t \in exbo\_at$
    grd2 : $exbo\_at \neq \varnothing \wedge$
           $exbo\_t > exbo\_a$
    grd3 : $exbo\_r =$
        $exbo\_req/(exbo\_t - exbo\_a)$
    grd4 : $paus\_stat = off$
  then
    act1 : $exbo\_r := 0$
    act2 : $rate := rate -$
        $exbo\_req/(exbo\_t - exbo\_a)$
    act3 : $exbo\_at := \varnothing$
    act4 : $exbo\_am := exbo\_req$
  end
Event EXBO_STOP ≙
  when
    grd1 : $exbo\_t \neq exbo\_a$
    grd2 : $exbo\_r =$
        $exbo\_req/(exbo\_t - exbo\_a)$
    grd3 : $paus\_stat = on$
    grd4 : $t \geq exbo\_a \wedge t \leq exbo\_t$
  then
    act1 : $exbo\_r := 0$
    act2 : $rate := 0$
    act3 : $exbo\_at := \varnothing$
    act4 : $exbo\_am :=$
           $exbo\_r * (t - exbo\_a)$
    act5 : $ba\_r := 0$
  end
```

The *EXBO_ON* and *EXBO_OFF* events satisfy *exbo-req-3*: *exbo_r* is assigned the requested extended bolus amount (*exbo_req*) divided by the duration of the extended bolus ($exbo\_t - exbo\_a$). *EXBO_ON* and *EXBO_OFF* may be disabled by the *PAUSE* event, derived from *grd3* and *grd4*, respectively. Because all requests shall be forbidden when the pump is paused (*IIP-req-5*), the stop for extended bolus infusion shall have two cases: one is that the extended bolus infusion is stopped naturally, *EXBO_OFF*, because time has reached the value of *exbo_t*; another is that the infusion processing is stopped by the pause, *EXBO_STOP*. The *exbo_at* set will become the empty set, when either the current time is equal to *exbo_t* or extended bolus is stopped by a pause. The *EXBO_STOP* event satisfies *IIP-req-6*, because the extended bolus is delivered evenly at rate *exbo_r*; when extended bolus is stopped by pause, the amount of extended bolus *exbo_am* that has actually been delivered can be calculated from $exbo\_r * (t - exbo\_a)$.

```
Event  PAUSE ≙
    when
        grd1 : $p\_stat = on$
        grd2 : $paus\_stat = off$
    then
        act1 : $ba\_r := 0$
        act2 : $exbo\_r := 0$
        act3 : $rate := 0$
        act4 : $exbo\_at := \varnothing$
        act5 : $paus\_stat := on$
    end
Event  RESUME ≙
    when
        grd1 : $p\_stat = on$
        grd2 : $paus\_stat = on$
        grd3 : $exbo\_at = \varnothing$
        grd4 : $exbo\_r = 0$
    then
        act1 : $ba\_r :=$
        $bpf(max(\{i | i \in ba\_t \wedge i \leq t\}))$
        act2 : $rate :=$
        $bpf(max(\{i | i \in ba\_t \wedge i \leq t\}))$
        act3 : $paus\_stat := off$
        act4 : $ba\_at :=$
        $ba\_at \setminus \{i | i \in ba\_at \wedge i \leq t\}$
    end
```

PAUSE and *RESUME* are typical of events such as *UNPRED_EVT_i* and *UNPRED_EVT_j* in the time pattern which appear in pairs. The *PAUSE* event sets *paus_stat* to on, and may disable lots of events such as *BA_CHANGE*, *EXBO_ON* and *EXBO_OFF* and set all the rates ($ba\_r, rate, exbo\_r$) to 0. The *RESUME* event has the same actions as the *ON* event for handling *ba_r* and *rate*. The *ba_at* set is recreated by deleting from the previous value of *ba_at* those elements smaller than or equal to the current time. Because *exbo_at* has already been set to $\varnothing$ in *PAUSE*, there is no action relevant to *exbo_at*.

```
Event  TICK_TOCK ≙
    when
        grd1 : $p\_stat = on$
        grd2 : $ba\_at \cup exbo\_at \neq \varnothing \Rightarrow t < min(ba\_at \cup exbo\_at) \vee paus\_stat = on$
    then
        act1 : $t := t + 1$
    end
```

This *TICK_TOCK* event is almost the same as the *TICK_TOCK* event in section 2.2. *grd 2* says the union of the basal time set and the extended bolus time set is not an empty set implies the current time is smaller than the minimum value of the union set, or the pause state is on, as the guard. In other words, only when the current time is smaller than the minimum of the global activation time set, can the time be increased by one.

Actually, there should be one more event here, which is *OFF* and satisfies *IIP-req-8*. The reason we do not include it here is that the actions of the *OFF* event are exactly the same as the *INITIALISATION* event. The only difference is that the *OFF* event has $p\_stat = on$ as guard.

4.3 Summary

During the process of creating this model in the Rodin Platform, a total of 79 proof obligations (POs) are generated, 65 of them being invariant preservation (INV) POs and 14 of them being well-definedness(WD) POs. 4 of the WD POs and 6 of the INV POs are semi-automatically proved, the other POs are automatically proved by the Rodin Platform. Because we did not apply any refinement steps to this model, POs corresponding to feasibility (FIS) POs, guard strengthening (GRD) POs and simulation (SIM) POs are not generated by the tool. More information about PO rules can be found in [2].

One of the reasons why we are not using any refinement steps on this model is that, in order to preserve the consistency of the system, the refined model shall be restricted by the previous model. In other words, the new events in the refined model must not change the values of variables in the previous or abstract model. For this time related control system, if we refine the model step by step, then each refinement stage may require events that change the time variable t. Because the time variable is used everywhere in the system and Event-B has this kind of feature to preserve the system consistency in refinement steps, we create the model at a single level of abstraction instead of using refinement. However, in the real project, our strategy to overcome the above shortcoming of our time pattern, which is reflected from the Event-B feature, is to introduce timing when the model has been refined to a very concrete level. This may not be an ideal way of going about specifying timing features for a critical system.

5 Conclusions and Future Work

This paper uses Event-B as the formal specification language to formalize a model of the IIP requirements. Although Event-B has useful features for building this model, the use of Event-B to handle the timing issues encountered in the IIP requirements is problematic.

The mechanisms outlined in this paper can solve two kinds of timing issues. One kind of timing issue results from introducing unpredictable events into the model (essentially these are externally invoked events). We modified an existing time pattern [7] by adding so called unpredictable events with new associated constraints and invariants to prevent the *TICK_TOCK* event from being inappropriately disabled. The action in *TICK_TOCK* makes the time tick incrementally and smoothly in order to enable us to capture the effects of unpredictable events. Another kind of issue is based on the modified time pattern; a large scale system may contain several events which share the same time constraints. Our solution is to classify the events into several categories; for each category we create its own independent time activation set and use the union of these sets as a global time set, instead of using only one global time activation set.

In the second half of the paper, we created a model for an IIP by applying the patterns mentioned above. From an examination of the model, we can see the pattern works well for the relevant time issues in this practical example.

Event-B is a mature formal specification language, but it still has some shortcomings. For example, there is no obvious way to check for completeness (as there is, for example, for table based specification [11,12]). When a developer is creating the models for big systems, missing some required events may easily happen. Only when we use third party plug-in tools to animate the model, can the errors resulting from the missing events potentially be detected. However, animation may not cover all the test cases, not to mention the problem that the animation tool itself may exhibit some faults. Therefore, we are trying to find other, more suitable formal specification languages that can overcome the disadvantages of Event-B; then we can combine them with Event-B and its tool to develop high quality software. Another disadvantage has been mentioned in the summary of section 4: to handle the timing issues we have to define the model in one step because of the variable restriction feature of refinement in Event-B and the effect that this has on refining time related aspects of the model. Although one solution for this problem is to use another variable name to describe time in a refinement step instead of using the old time variable name in the refined model, the refinement of the model will include both the new time variable and the old time variable, which makes the model look unintuitive and, from the point of view of analysis, more complex and harder to analyze. We would be interested to see if this problem can be solved by the Event-B community.

References

1. Zhang, Y., Jones, P.L., Jetley, R.: A Hazard Analysis for a Generic Insulin Infusion Pump. Journal of Diabetes Science and Technology 4, 263–283 (2010)
2. Abrial, J.-R.: Modelling in Event-B: System and Software Engineering. Cambridge University Press (2010)
3. Xu, H.: Model Based System Consistency Checking Using Event-B. Masters thesis, McMaster University (2011)
4. Abrial, J.-R.: The B-book: assigning programs to meanings. Cambridge University Press (1996)
5. Cansell, D., Méry, D.: The Event-B Modelling Method: Concepts and Case Studies. In: Bjorner, D., Henson, M.C. (eds.) Logics of Specification Languages, pp. 47–152. Springer (2008)
6. Abrial, J.R., Butler, M., Hallerstede, S., Hoang, T.S., Mehta, F., Voisin, L.: Rodin: an open toolset for modelling and reasoning in Event-B. International Journal on Software Tools for Technology Transfer (STTT) 12, 447–466 (2010)
7. Cansell, D., Méry, D., Rehm, J.: Time Constraint Patterns for Event B Development. In: Julliand, J., Kouchnarenko, O. (eds.) B 2007. LNCS, vol. 4355, pp. 140–154. Springer, Heidelberg (2006)
8. Rehm, J.: A Methods to Refine Time Constraints in Event B Framework. In: Merz, S., Nipkow, T. (eds.) Automatic Verification of Critical Systems - AVoCS 2006, Nancy, France, pp. 173–177 (2006)
9. Lynch, N., Vaandrager, F.: Forward and Backward Simulations - Part II: Timing - Based Systems. In: Meyer, A.R. (ed.) Information and Computation, vol. 128, pp. 1–25. Elsevier (1995)
10. Alur, R., Dill, D.L.: A Theory of Timed Automata. Theoretical Computer Science 126, 183–235 (1994)
11. Jin, Y., Parnas, D.L.: Defining the Meaning of Tabular Mathematical Expressions. In: Bergstra, J.A. (ed.) Science of Computer Programming, vol. 75, pp. 980–1000. Elsevier (2010)
12. Janicki, R., Wassyng, A.: Tabular Expressions and Their Relational Semantics. Fundam. Inf. 67, 343–370 (2005)

On the Safety of Electronic Medical Records

Jens H. Weber-Jahnke and Fieran Mason-Blakley

University of Victoria, Victoria BC V8W 3P6, Canada
jens@acm.org, fieranmason@gmail.com
http://simbioses.ca

Abstract. Information and communication technology is rapidly transforming modern health care systems. Electronic Medical Records (EMRs) systems have replaced traditional forms of storing, processing, interpreting and exchanging patient health information in many health care organizations. However, an increasing number of concerns are raised about the quality of EMR systems and industry regulators are pondering ways to ensure safer health information technologies. This paper discusses fundamental concepts associated with the safety of EMR systems, describes current approaches to regulating the industry, and discusses limitations of traditional safety engineering methods with respect to their application to EMR systems. We then present a domain-specific adaptation of Leveson's system-theoretic model STAMP for safety engineering of EMR systems and demonstrate its application with a real-world case study.

Keywords: electronic medical records, safety, eHealth, system theory.

1 Introduction

Electronic Medical Records (EMRs) play an important and increasingly critical role in modern health care delivery. They are computer-based and have replaced or are about to replace traditional paper-based health information management practices. EMRs have been viewed as contributing to solutions that ensure the long term sustainability and quality of modern health care systems, while having a dampening effect on the rapidly rising costs in this domain [24]. Since medical data in EMRs are readily computer-interpretable, these systems also have the potential to accelerate medical research and benefit evidence-based public health initiatives. Industrialized countries have invested significant resources in the development and adoption of EMR technologies and infrastructures.

However, the increasing diffusion of EMR technologies in medical practice has been met with growing concerns about their safety implications. While researchers have initially published predominantly about (potential) safety benefits of EMRs, observed in experiments with prototypes in carefully controlled environments, voices have become far more critical when it comes to experiences with commercial EMR products in day to day health care practices [3,17,6]. Caregivers and practitioners have alerted the industry and governments about unforeseen and potentially dangerous 'side-effects' of EMRs [30]. There is mounting pressure on regulators to put in place mechanisms that assure the safety of health systems using EMRs [28,31,10].

While researchers and practitioners have called for mandatory regulation of health information technology and regulators have started to act, confusion and disagreement exists about what approach should be taken. Moreover, there has been significant push-back by the industry about concerns that the application of traditional regulatory controls would be an ineffective burden to software manufacturers. But how can we engineer EMR systems for safety? In this paper, we seek to inform this discussion by characterizing fundamental properties associated with the safety of EMRs and contrasting them with safety properties of other devices. We will then discuss the limitations of existing safety engineering methods and describe an application of a systems-oriented safety model to the analysis of hazards in EMR systems. The contributions of this paper are (1) a characterization of fundamental properties associated with EMR system safety, (2) a discussion of current regulatory approaches to EMR safety under the medical devices regulations, (3) a discussion of the limitations of traditional safety engineering methods, and (4) the description and application of a system-theoretic safety engineering method based on Leveson's STAMP model [20].

The rest of this paper is structured as follows. The next section defines important concepts related to safety in the health care domain, in general, and the concept of EMR safety, specifically. Sec. 3 gives an overview of current approaches on regulating the EMR industry for safety and highlights several existing issues and challenges. In Sec. 4, we argue the limited applicability of traditional safety engineering methods and describe a system-theoretic model based on Leveson's STAMP that can overcome these limitations. We demonstrate the use of the proposed model for analysing a real world accident involving a serious medication overdose. The last section summarizes the main results of this paper and contains concluding remarks about future research directions. We reference related work throughout the paper.

2 Safety in Health Care

2.1 General Concepts

Safety is commonly viewed as a system property and has been defined as "freedom from those conditions that can cause death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment" [1]. Devices are referred to as *safety-critical* if they are essential for the safe operations of a system, i.e., if their failure alone could result in death or system loss. Otherwise, devices may be called *safety-sensitive* if they contribute to a safety-critical functions.

2.2 Adverse Events and Medical Errors

Considering the above definition of safety, the medical domain can be considered inherently unsafe, since it necessarily deals with injury and death. While there is broad consensus that safety must be considered a relative rather than an absolute [21], reasoning about the safety of EMRs may indeed be more difficult because of this inherent characteristic of the health care domain. The term *'adverse event'* is commonly used in medicine to refer to an instance of patient harm not due to the natural course of a disease or illness [23]. Adverse events are further categorized as preventable or non-preventable. Examples for non-preventable adverse events are harmful side-effects of medically necessary drugs or treatments such as radiation therapy. It is often not easy to distinguish preventable adverse events from patient harm due to 'natural causes'. Nevertheless, the number of recorded adverse events is significant. According to a study of the Institute of Medicine (IOM), more U.S. citizens are killed by adverse events than by highway accidents or breast cancer [15].

The concept of preventable adverse events is akin to that of *medical errors*, which has been subject to much debate and is often (but not always) associated with human actions. Bucknall and Botti define that a medical error occurs "when persons fail to complete an action as planned or intended (an act of omission), or they use an incorrect plan to achieve an aim (an act of commission)" [7]. Notably, not every medical error may lead to an adverse event. A medical error that did not contribute to any adverse events is called a *latent medical error*, *close call* or *near miss* [23]. Moreover, not all adverse events have their cause in medical errors (when defined as failed actions of human caregivers). For example, adverse events caused by malfunctioning medical devices (such as a failing pacemaker) will not normally be designated as a caregiver error. Moreover, modern health care services are increasingly team-based, specialized and collaborative, that certain 'collective errors' (e.g., communication failures) cannot easily be pinned down to failures of individuals [18].

2.3 Actor Perspective vs. System Perspective

As the preceding discussion indicates, the concept of medical safety can be discussed from different perspectives and the choice of perspective has a major influence on how safety is handled in the medical domain. The actor perspective focusses on errors made by individuals and is based on the cognitive theory of actions, which characterizes each action as a process of seven stages: establishing a goal, forming an intention, specifying an action, executing the action, perceiving the system state, interpreting the system state, and evaluating the system state [32]. Medical errors may be made in any of these stages. Safety initiatives focus on minimizing the likelihood of medical errors by improving on the training of the actors and the tools, methods, and processes they apply (including EMRs). For example, a commonly reported type of medical error is associated with inattention and interruptions of caregivers. This may happen when a caregiver who is in the process of performing an action with an EMR is interrupted, e.g., through a phone call or a personal consultation [22]. After the interruption, the caregiver's internal representational model of the action may have degraded or, if the interruption also required interaction with another patient's EMR, the caregiver may mix up aspects of different actions. Approaching safety from an actor perspective could for example attempt to decrease the chance of medical error by designing the EMR software to provide cognitive support to caregivers, helping them to safely recover from interruptions.

In contrast, the system perspective would take a more holistic approach to safety, trying to identify and improve on the potentially multiple factors that may contribute to medical errors [25]. Consequently, given the above example, a system perspective safety initiative may also seek to restructure the health care environment to decrease the frequency of interruptions in the first place.

2.4 Describing Medical Errors and EMR Errors

Consistent classification of errors is an important prerequisite for reporting, researching and mitigating safety hazards. The effectiveness of reporting systems, such as the U.S. Food and Drug Administration's Manufacture and User Facility Device Experience (MAUDE) database[1] and the ECRI Medical Device Safety Report system,[2] depends on the use of a consistent nomenclature and structure.

Various taxonomies for classifying medical errors have been proposed, e.g., [8,23,26]. Perhaps the most comprehensive taxonomy was developed by the Joint Commission on the Accreditation of Health Care Organizations (JCAHO) and comprises over 200 classes under five 'root nodes', defining *impact*, *type*, *domain*, *cause* and *prevention/mitigation* of each class of medical error [8]. While these taxonomies provide a starting point for a discussion of EMR-related hazards, it has been pointed out that EMR safety concerns are broader than the traditional notion of medical errors. Early work by Smith categorizes the types of patient harm in relative order of decreasing severity: (1) *direct physical harm* (e.g., surgery on the wrong side because of flipped information), (2) *indirect physical harm* (e.g., delays of patient treatment due to system unavailability), (3) *mental suffering and distress* (e.g., due to wrongfully associating worrisome test results with healthy patients), (4) *financial loss* (e.g., compensation claim disputed because of incorrect information about past services), (5) *loss of personal reputation or livelihood* (e.g., data breach in EMR divulges embarrassing or damaging personal health data), and (6) *loss of opportunity to improve care* (e.g., inability to economically research and optimize EMRs may indirectly harm patients through suboptimal care) [29].

In a more recent work, Philipps and Gong suggest a structured nomenclature to describe 'EMR errors', a concept they define as "*any incorrect data or faulty functionality, any aspect of the* [*EMR*] *system not functioning according to design specification or user requirements, any incorrect design specification or user requirement, and any significantly suboptimal usability*" [23]. Philipps and Gong point out the difficulty of crossing the chasm between the medical and the engineering domain: general classification schemes for medical errors are too vague to describe EMR errors with sufficient technical depth, and technical terminologies are not very meaningful for users who operate in a clinical context.

[1] `http://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfMAUDE/search.CFM`
[2] `http://www.mdsr.ecri.org/`

3 Regulatory Approaches to Increasing EMR Safety

Driven by alarming concerns about the safety of EMRs, several jurisdictions have considered extending existing medical device regulations to cover EMR software. Certain types of EMR-like software that directly interfaces with medical device hardware has long been subject to regulation, for example Laboratory Information Systems (LIS), Radiology Information Systems (RIS), and Picture Archiving and Communication Systems (PACS). However, several jurisdictions have recently begun to expand their regulatory regimes to also cover more general types of EMRs, e.g., Canada [31], Europe [27] and the U.S. [28,9].

In 2009, Health Canada notified industry about their ruling to consider patient management software (including EMRs) medical devices [2]:

> *"Patient management software having the ability to affect the diagnosis or treatment of a patient fits the definition of a medical device and must comply with the requirements of the Regulations."*

However, Health Canada withdrew that notice a few months later after consultation with industry and replaced it with a new notice avoiding the term 'patient management software' and publishing an FAQ that answers the question *"Are Electronic Medical Records (EMRs)* [...] *medical devices?"* as follows:

> *"A software product that simply replaces a patients paper file does not meet the definition of a medical device if it is only intended to store and view patient information* [...]*."*

Canada's current approach to considering EMRs that 'simply replace' a patient's paper file as lower risk (not a medical device) while classifying other software such as clinical decision support systems as higher risk (medical device) is indicative of the failure of some regulatory bodies to appreciate a system-theoretic perspective of EMR safety. Indeed, in an information-intense discipline like health care, the availability and quality of data (facts) about patients plays a crucial role for their safety. In fact, it can be argued that 'intelligent' decision support software has a lower overall risk (i.e., they are *'safety sensitive'*), compared to the fundamental data storage, view, retrieval and communication functions of EMRs (i.e., they are *'safety critical'*), since clinicians do not normally assume that computer-based decision support functions are error-free. In other words, the hazardous impact of errors in higher-level software-based functions that perform analysis on basic patient data is generally lower than the hazardous impact of errors in the software that manages the basic patient data. As Holden states it, "*information systems (manual or computerized), for instance, critically determine clinicians awareness of the patient situation* [...]." [11].

The U.S. FDA takes a different, arguably more sensible approach by recognizing the criticality of all clinical EMRs and claiming jurisdiction under the medical devices regulation, even if it is not yet clear how the regulation should best be enforced for different types of systems [28]. The FDA has recently issued a new rule on Medical Data Device Systems (MDSS) to control the most simple types of data systems, i.e., *"hardware or software products that transfer, store, convert formats, and display medical device data"* [9]. According to this definition, EMRs with the capability to transmit data that originate from medical devices (such as blood pressure monitors) electronically could be considered MDSS.

4 Methods for Engineering Safer EMR Systems

4.1 Limitations of Event-Based Methods

Many traditional methods applied for the purpose of analyzing hazards of software systems are event-based. Event Trees model potential chains of events with the purpose of identifying hazards (forward direction), while Failure Trees model possible event chains that may have contributed to a failure with the purpose of identifying the *root cause* of a failure. Leveson discusses the limitations of event-based methods with respect to engineering complex socio-technical systems [19]. She criticizes that the choice of events to be considered in such models is subjective and the causality relationships considered are often too simplistic. Moreover, event-based methods focus on identifying root causes and assigning blame – a process that is generally problematic in socio-technical systems and often terminates when a human operator can be identified who could have prevented the accident, without attempting to understand why he acted the way she did.

The focus of event-based methods on assigning blame is particularly problematic in health care and their application may even contribute to making these systems less safe. This is in part because of a legal doctrine called the "learned intermediary rule", which has served as a legal defense for the health software industry and effectively shifts liabilities to the user, given that the vendor provided warnings about potential safety hazards [16]. This doctrine has caused a difficult socio-technical conundrum, as it encourages EMR software vendors to develop systems that generate lots of alerts, which in turn have to be dismissed by clinicians, while each overriding action is tracked and stored by the EMR software [4,13]. This conundrum has caused frustration among clinicians.[3]

Another limitation of event-based methods is that the analysis tends to stop with the proximal cause that eventually triggered the accident, rather than attempting to identify the systemic reasons behind a general deterioration of safety margins [19]. This limitation is especially relevant to our application domain, since the safety of an EMR to a large degree depends on quality of the contained data, which tends to deteriorate over time. For example, consider the following excerpt of an EMR safety report submitted to the FDA MAUDE (cf. Sec. 2.4)

> **Report: MW5014248. Outcome: Injury.** Vital data on critically ill pt was missing from the [EMR] over a duration of several days. [...] The pt had a double identity in the [EMR]. Spuriously, the data transferred to the other identity, that was not present on users' pt lists. The [EMR] did

[3] see `hcrenewal.blogspot.com/2009/03/health-care-information-technology.html`

not provide indication that patients had two or more listings, nor were the two identities combined despite same names and vital statistics.

Fig. 1 shows an excerpt of a possible fault tree for this failure. The error report submitted by the user suggests finding a root cause in the fact that the EMR software did not alert the user about the presence of multiple records with similar data (cf. the event marked with a black star in Fig. 1). A vendor statement responding to this reported incident may however assign blame to the user for not searching for patient records with the same name (and similar data) prior to entering and retrieving the prescription, respectively (cf. events marked with a white star).

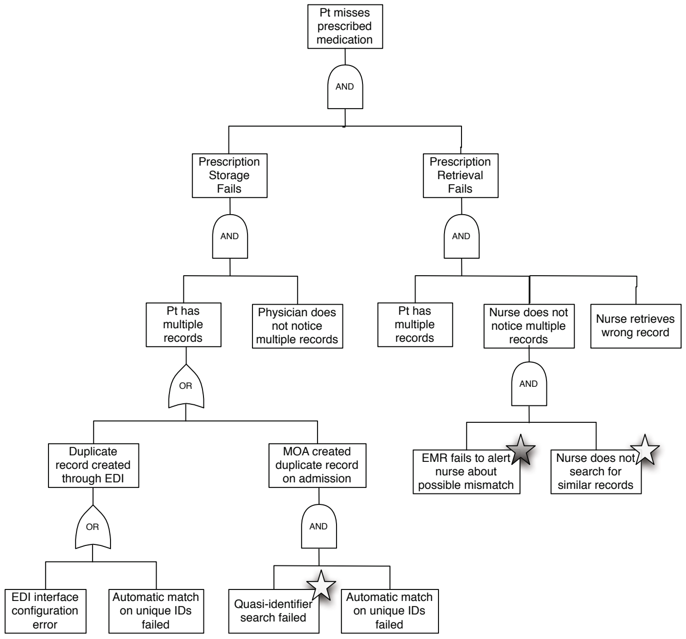

Pt misses prescribed medication
AND
Prescription Storage Fails
Prescription Retrieval Fails
AND
AND
Pt has multiple records
Physician does not notice multiple records
Pt has multiple records
Nurse does not notice multiple records
Nurse retrieves wrong record
OR
AND
Duplicate record created through EDI
MOA created duplicate record on admission
EMR fails to alert nurse about possible mismatch
Nurse does not search for similar records
OR
AND
EDI interface configuration error
Automatic match on unique IDs failed
Quasi-identifier search failed
Automatic match on unique IDs failed

Fig. 1. Partial Fault Tree for Patient Mismatch Hazard (FDA Report MW5014248)

However, both candidates for root causes appear unsatisfactory. In the first case, the vendor may point to conceptual difficulties of automatically identifying duplicate records in the absence of unique identifiers. In the second case, clinicians may rightfully complain that they are not able to operate efficiently if they have to check for potential health record similarities each time they write a prescription or retrieve one. Even if blame is assigned to both system components (EMR and clinician) and controlling measures are put in place in the technology as well as in the prescription process, the overall result for patient safety still remains unsatisfactory. This is because the overall likelihood of the described adverse event is bound to increase over time if data quality continues to deteriorate, i.e., more spurious records are created over time.

It is more appropriate to identify the root cause for the ‘multiple patient record’ hazard as the absence of an agreement within the health care system on a common unique identification for patients. Even if nation-wide unique patient IDs should be deemed out of reach in the short term, failure to put in place regular and proactive measures to counteract the deterioration of EMR data quality with respect to potential multiple patient records can be considered a more plausible and realistic ‘root cause’ than the events contained in the above fault tree. However, the lack of a direct causal link between the reported adverse event and a potentially regularly occurring data quality improvement measure (e.g., weekly screening for duplicate records) means that such a root cause will normally not show up in event-based models.

Data quality hazards in EMR systems are not limited to information about patients but may also pertain to meta data. A common hazard is caused by the deteriorating quality of terminologies. Consistent use of terminologies is critical for health information processing, query and retrieval. For this reason, the International Health Terminology Standards Development Organization (IHTSDO) and other organizations have been maintaining *controlled terminologies* to describe health related concepts and relationships. For example, the Systematized Nomenclature of Medicine–Clinical Terms (SNOMED CT) consists of a polyhierarchy of over 300,000 concepts and close to one million relationships. Unfortunately, the resulting terminologies are inherently bound to be complex yet incomplete. Therefore, EMRs necessarily need to provide their users with means to customize and extend locally used, so-called *interface terminologies*. Clinicians who may not readily find the terms needed to describe a particular medical condition in a patient’s record are able to define and use a new term. Unfortunately, clinicians may not always use the right spelling to look up controlled terms and define new terms. Studies have shown that this leads to a significant deterioration of data quality over time [5]. Even without the element of linguistic errors, humans are used to applying different terminologies depending on their background. Moreover, even controlled terminology standards are not static but evolve dynamically. For example, IHTSDO releases an update to SNOMED CT every six months. Without the instigation of measures to compensate for the resulting deterioration of data quality, the safety margins of EMR systems will decrease over their lifetime.

4.2 System-Theoretic Approaches

Among the many factors that influence the safe operation of an EMR system, only few are determined by the design of *technical* artifacts alone. System-theoretic approaches to safety engineering attempt to take non-technical as well as technical factors into account when reasoning about the safety of complex socio-technical systems. As illustrated in the previous section, EMR systems must be considered as dynamically evolving (and adapting) rather than being static. A direct implication of considering dynamic systems is that safety cannot be established in a one-shot analysis activity, but rather must be treated as a continuous process. As Leveson puts it: “*... the struggle for a good safety culture* *will never end because it must continually fight against the functional pressures of the work environment.*" [20] The realization that EMR safety depends on our ability to control dynamic socio-technical systems has important ramifications for regulators who are pondering the proper choice of controls to put in place for health information systems (cf. Sec. 3). The traditional approach of medical device regulators to (1) conduct post-market surveillance for low-risk devices, (2) require (additional) certifications on good manufacturing processes for medium risk devices, and (3) add pre-market safety approvals for high risk devices does not seem appropriate for complex socio-technical systems such as EMRs.

Theories of dynamic systems usually incorporate a notion of *control loops*, where a controller (or multiple controllers) monitors the state of a process and influence that process to approximate a defined optimization objective. Leveson has adopted this theory in her safety engineering model STAMP (*Systems-Theoretic Accident Model and Processes*) [20]. STAMP reformulates the traditional safety problem of *preventing accidents* into a problem of *constraining performance*. This perspective is particularly helpful in health care, since it is not trivial to distinguish adverse events (accidents) from patient harm due to 'natural causes' (cf. Sec. 2.2). This focus on process performance has received much recent support by health care researchers. As Karsh et al. state:

> "Patient safety is the product of how and how well health care providers (HCPs) perform cognitive work processes. There is no direct way to reduce errors or harm. Instead, errors are reduced when the conditions of work (i.e., the work system [including the EMR]) positively shape the way that HCPs perform cognitive work; harm is reduced when the conditions of work allow HCPs to perform well under challenging or disruptive conditions. The primary goal should not be to reduce error or harm. The goal should be to design work systems [including EMRs] that support and enhance work process performance. Error reduction and harm prevention will follow in turn." [14]

Holden recently published a comprehensive study demonstrating the cognitive performance-altering effects of EMRs [11].

Control loops in complex socio-technical systems, such as the health care system, must be considered at different hierarchical levels both, for system *development*, as well as during system *operations* [20]. Each level has its own set of constraints to achieve the desired system performance objectives. For example, controlling an individual clinician's work system may be considered at the lowest level, while the operations of an entire health care organization (clinic or hospital) may be considered as the next level, and so on until we reach the level of industry regulators and governments. Communication links exist between the different control loops in the hierarchy.

4.3 A System-Theoretic Model for EMR Safety

In this section, we describe how STAMP can be applied to reason about the safety of EMR systems. Fig. 2 shows a domain-specific model (here referred to as

STAMP /EMR) that can be used to identify and classify safety hazards. In this model, the patient's health takes the role of the '*controlled process*' in the generic STAMP framework. The controller has two components, the physician and the EMR. Both controller components have a model of the patient's health process. The physician observes and controls patient's health process through the computer-based EMR, while the EMR receives updates about the patient's health status from medical device sensors (e.g., vital signs, blood work, etc.). The orders entered in the EMR are executed by actuators that influence the patient's health process. These actuators may be automated, partially automated, or manual. A variable-flow rate, computer-controlled analgesic infusion pump that is connected to the patient is an example for an automated actuator, while a nurse who deploys a new IV drip infusion using a drug compound created with a computer-controlled compounder device can be considered a partially automated actuator.

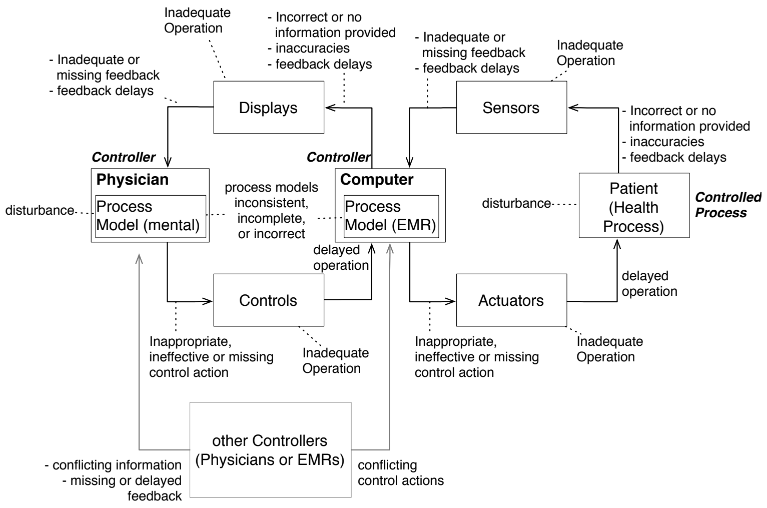

Inadequate Operation
- Incorrect or no information provided
- inaccuracies
- feedback delays
- Inadequate or missing feedback
- feedback delays
Inadequate Operation
- Inadequate or missing feedback
- feedback delays
Displays
Sensors
- Incorrect or no information provided
- inaccuracies
- feedback delays
Controller
Controller
Physician
Process Model (mental)
process models inconsistent, incomplete, or incorrect
Computer
Process Model (EMR)
disturbance
disturbance
Patient (Health Process)
Controlled Process
delayed operation
Controls
Actuators
delayed operation
Inappropriate, ineffective or missing control action
Inadequate Operation
Inappropriate, ineffective or missing control action
Inadequate Operation
other Controllers (Physicians or EMRs)
- conflicting information
- missing or delayed feedback
conflicting control actions

Fig. 2. STAMP Model Applied to EMR Systems

Of course, the control loop in Fig. 2 is idealized and purposefully hides some of the complexity of the health care process. For example, physicians often observe and control the patient's health process not only indirectly through the EMR, but also more directly by applying medical devices themselves. Moreover, the depicted controllers are of course part of a hierarchy of higher level control loops, such as the hospital safety regulations, the health authority, etc. Nevertheless, even the simplified model given above can be used to identify and analyze many types of hazards associated with the clinical use of EMRs. The types of hazards are either related to inconsistent, incomplete or incorrect process models (physician or EMR), or they pertain to communication problems within the control loop, e.g., inadequate, missing or delayed feedback on control actions performed (cf. annotations in Fig. 2). Moreover, other controllers (other physicians and other computer-based EMRs) may interfere with conflicting information or control actions.

4.4 Applying the STAMP/EMR Model in Practice

STAMP/EMR can be used pro-actively for the identification of hazards during the process of designing and evolving safer systems, as well as reactively during the investigation of the reasons behind accidents that have occurred. Leveson has developed two techniques that guide the application of STAMP in these situations: STPA (System Theoretic Process Analysis) is a STAMP based technique for designing safe systems, while CAST (Causal Analysis based on STAMP) is a technique for investigating accidents that have occurred [20]. In this section, we show the application of CAST and STAMP/EMR for investigating the reasons behind a serious medication error that occurred at the NewYork Prebyterian Hospital and was later reported on by Horsky et al. [12]. The following case summary of proximal events has been adapted from Horsky's paper:

Case Summary - Proximate Events. On Saturday morning, the physician on duty at the pulmonary service unit (Physician A) examined routine laboratory test results of an elderly patient that showed a serum KCl of 3.1 mEq/L and correctly diagnosed the patient as hypokalemic (low potassium value). She ordered an intravenous (IV) bolus injection of 40 mEq of KCl (potassium chloride) to be delivered over 4 hours. Immediately after entering the order, Physician A realized that the patient already had an IV fluid line inserted and therefore decided to deliver the KCl as an additive to the currently running IV fluid, in order to make the interaction less painful for the patient. Consequently, she entered a new order to infuse 100 mEq of KCl in 1 L of drip solution at the rate of 75 mL/hr. Intending to cancel the preceding order for an injection, Physician A inadvertently discontinued a similar order that was entered by another clinician two days ago. Later, Physician A received a call from the pharmacy, notifying her that the dose of the new drip order exceeded the maximum limit allowed by hospital policy. She consequently discontinued the drip order and wrote a new drip order for fluid medicated with 80 mEq/L KCl. However, this order was not entered correctly: the intended volume limit of 1L was not properly specified in the order, which lead to a continued administration of KCl over 36 hours, delivering a total of 216 mEq KCl (36 x 75 mL = 2.7 L; 2.7 x 80 mEq = 216 mEq). Including the first bolus of 40 mEq KCl, the patient had inadvertently received a total of 256 mEq KCl over 36 hours.

On Sunday morning, Physician B took over the office from Physician A, who notified the incoming officer to check the patients KCl level. Physician B reviewed the patients most recent available serum potassium value, which was 3.1 mEq/L. Even though the date and time (i.e., Saturday, the previous morning) of the result were displayed on the EMR screen, Physician B did not realize that the test result was in fact from before the last potassium repletion and acted as though the patient was hypokalemic. He ordered an additional 60 mEq KCl to be given as an IV injection, even while the previous potassium drip was still running. Order entry logs also indicated that another 40 mEq KCl IV injection was ordered by Physician B about 30 minutes later, but there is no clear evidence from other sources that it was in fact administered to the patient. As a result, the patient received a total of 316 mEq KCl over 42 hours. On Monday morning, when the KCl laboratory values were checked, the patient was found to be severely hyperkalemic (serum K level at 7.8 mEq/L).

In their case report entitled "*Comprehensive Analysis of a Medication Dosing Error Related to CPOE*" [12] Horsky et al. included several recommendations for safety improvements (summarized later in this section) but do not provide a systematic technique for their elicitation. In the following we show that (a) STAMP/EMR and CAST can provide such methodological guidance and applying this method will identify the recommendations documented in Horsky's case report, and (b) applying STAMP/EMR and CAST will yield additional recommendations that capture safety concerns not considered in the original investigation. We will now successively iterate through the nine-step CAST technique using our case study. We refer to [20] for a deeper exposition of CAST.

Step 1 - ***Identify the System Hazards Involved in the Loss.*** The first step in the analysis process has the objective of identifying the hazards involved in the loss. The main hazard is clearly that (H1) "*the patient receives a potassium overdose*". A secondary hazard can be considered (H2) as "*the intervention causes painful side-effects for the patient*". Note that in CAST we are only concerned with those hazards that were actually involved in the accident. Other hazards may exist (e.g., "*patient fails to receive medication*") but are not considered in the analysis of a concrete accident.

Step 2 - ***Identify Safety Constraints Addressing the Hazards.*** The second step seeks to identify all system level safety constrains associated with the hazards identified under Step 1. They are (C1) "*the patient must not be given a KCl overdose*", (C2) "*potassium intervention must stop if patient's KCl levels are too high*", and (C3) "*KCl should be repleted using a drip line if such a line pre-exists*". Note that C3 is associated to H2 while C1 and C2 correspond to H1.

Step 3 - ***Document the Safety Control Structure in Place.*** The goal of this third step is to document the roles and responsibilities of each system component in enforcing the safety constraints and the relevant feedback provided to help them. Fig. 3 shows the safety control structure for our case study. Both physicians are responsible with enforcing C1, C2 and C3. The pharmacy enforces a limit on the maximum concentration of a drip fluid KCl medication. The EMR injection order controls enforce volume limits (as entered by the physicians). The EMR drip order controls enforce medication duration limits (as entered by physicians) or, if no limit is entered, medication is discontinued after seven days. Moreover, the drip order controls can be used to enforce limits on the bag size. Finally, the EMR biochem test display highlights out-of-range values. The interaction links in Fig. 3 show that physicians do not receive feedback from the controls about the implementation of their control actions.

Step 4 - ***Determine the Proximate Event Leading to the Accident.*** There are about 20 proximate events to be considered in the accident. They have been documented in Fig. 4 using sequence numbers. In some cases the exact sequencing of events could not be determined by the committee investigating the

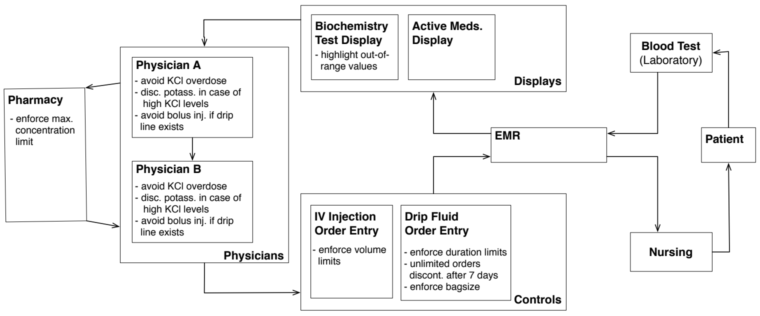

Biochemistry Test Display
- highlight out-of-range values
Active Meds. Display
Displays
Blood Test (Laboratory)
Physician A
- avoid KCl overdose
- disc. potass. in case of high KCl levels
- avoid bolus inj. if drip line exists
Pharmacy
- enforce max. concentration limit
Physician B
- avoid KCl overdose
- disc. potass. in case of high KCl levels
- avoid bolus inj. if drip line exists
Physicians
EMR
Patient
IV Injection Order Entry
- enforce volume limits
Drip Fluid Order Entry
- enforce duration limits
- unlimited orders discont. after 7 days
- enforce bagsize
Controls
Nursing

Fig. 3. Safety Control Structure for NewYork Prebyterian Hospital Accident

accident. Also, as mentioned above, it is not clear whether the last injection order by Physician B resulted in an actual administration of a medication dose.

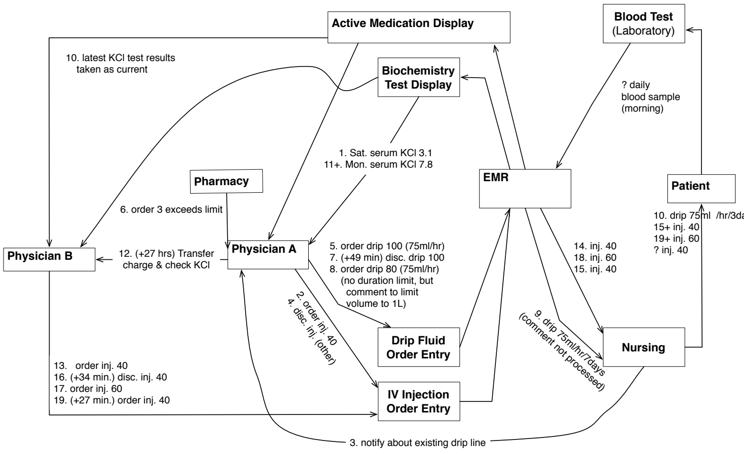

Active Medication Display
Blood Test (Laboratory)
10. latest KCl test results taken as current
Biochemistry Test Display
? daily blood sample (morning)
1. Sat. serum KCl 3.1
11+. Mon. serum KCl 7.8
Pharmacy
EMR
Patient
6. order 3 exceeds limit
10. drip 75ml /hr/3days
15+ inj. 40
19+ inj. 60
? inj. 40
Physician B
12. (+27 hrs) Transfer charge & check KCl
Physician A
5. order drip 100 (75ml/hr)
7. (+49 min) disc. drip 100
8. order drip 80 (75ml/hr) (no duration limit, but comment to limit volume to 1L)
14. inj. 40
18. inj. 60
15. inj. 40
2. order inj. 40
4. disc. inj. (other)
9. drip 75ml/hr/7days (comment not processed)
Drip Fluid Order Entry
Nursing
13. order inj. 40
16. (+34 min.) disc. inj. 40
17. order inj. 60
19. (+27 min.) order inj. 40
IV Injection Order Entry
3. notify about existing drip line

Fig. 4. Sequence of proximate events leading to the loss

Step 5 - ***Analyze the Accident at the Physical System Level.*** This step investigates failures of physical components and flaws in their control, interaction and coordination that may have contributed to the loss. No such failures have been reported in our case study.

Step 6 - ***Analyze the Accident at Higher Levels of Control.*** This step successively considers each higher level of the control structure to identify why they allowed the accident to occur or contributed to its occurrence. For each level of control, we consider (1) which responsibilities for enforcing the identified safety constraints were assigned, (2) should have been assigned, and (3) whether enforcement actions were exercised sufficiently and correctly. For any flawed human decision, we consider the information available to the decision maker compared to the information that should have been available to her. Contextual factors of the decision-making are also considered, such as behaviour-shaping mechanisms, value structures, disturbances etc.

Let us consider the physicians first. Physician A is missing important information: she is unaware of the patient's existing IV drip connection; she confuses her injection order with an expired order and assumes that her active order was discontinued; and she believes that doses on drip orders can be limited by volume. (She uses the 'Total Volume' field to enter 1L, assuming that this would limit the dose, however, the field was merely designed to specify the size of the infusion bag.) She believes that directions entered in the 'comment' field of the order entry screen. (She also uses the 'comment' field to specify directions on the 1L volume limit. However, the free text comments are not normally considered during order processing.) She assumes that the KCl drip order stops after 1 L.

These mental model flaws lead to several inadequate decisions and control actions: She initially orders a bolus injection instead of the less painful drip; She discontinues the wrong order; She misuses the '*Total Volume*' and commenting fields in the drip order entry screen of the EMR for limiting the medication order; She fails to communicate to the incoming Physician B details about her standing drip order intervention.

Physician B is also unaware of the patient being connected to a drip line, as well as that the drip infusion is medicated with KCl. Moreover he assumes that the latest blood test results are current. Finally, he is unaware of the status of his last medication injection order (whether it has been administered).

The EMR (and its displays and controls) have also contributed to the accident. The EMR screen that is normally used by clinicians to display active medications does not show medicated drip infusions connected to the patient, the display of laboratory test results shows dates but does not indicate whether the latest result can be considered 'current', the two control screens used for ordering medication are very similar in appearance, but very different in their behaviour: injection orders are limited by volume, while drip infusion orders must be limited by flow rate and duration (and cannot be limited by volume), the naming of the field 'Total Volume' is ambiguous and may mislead physicians to assume that they can use it for limiting medication doses, the 'comment' field may lead clinicians to specify dosage constraints.

Other components in our safety control structure could also have prevented the accident. For example, the nurse giving the KCl injections could have detected that the patient was already on an active KCl drip medication (check the bag), and the pharmacy could have checked total dosage limits in addition to maximum dosage concentrations. Moving through the (non-physical) components of our safety control structure (Fig. 3), we can identify the following missing constraints:[4] *Nurse*: (*1) "must not inject KCl in patients already connected to a KCl medicated drip fluid line" and (*2) "must read and acknowledge

[4] Note that these constraints have been identified based on the full accident report by Horsky et al.. Some of them may not be identifiable based on the abbreviated summary of proximate events provide in this paper.

comments on medication orders"; *EMR*: (*3) "must display order comments to nurses", (*4) "must validate that there is no active KCl drip fluid order for a KCl injection order to proceed", (*5) "must audit ordering history of physicians and raise alert when pattern of repeated orders and cancellations are detected"; *Drip Fluid Order Entry*: (*6) "must enforce volume limits" and (*7) "must avoid confusion between total volume constraints and bag size constraints"; *Controls* (in general): (*8) "must avoid confusion between injection orders and drip orders" and (*9) "must indicate that comments cannot be used for dosage constraints"; *Physicians*: (*10) "must confirm that the 'latest' lab results are still 'current'", (*11) "must validate whether a cancelled injection order has been administered prior to converting into a drip order"; *Active Meds Display*: (*12) "must display active medicated drip fluids"; *Biochemistry Test Display*: (*13) "must highlight 'dated' test values".

Step 7 - ***Examine Coordination and Communication Issues.*** Even the abbreviated summary of the accident we give in this paper shows that lack of coordination and communication was a major contributor to the accident. We can identify the following coordination/communication flaws (+1) "physicians received insufficient feedback on whether (and when) their medication orders were administered", (+2) "the existence of (medicated) drip fluid lines were not well communicated to physicians" and (+3) "the currency of lab test data was not well communicated". These three identified coordination and communication flaws give rise to more detailed communication requirements between the components in our safety control structure.

Step 8 - ***Dynamic Changes in the Safety Control Structure.*** As described earlier, socio-technical systems are dynamic and system changes over time may have contributed to the accident. In our case, study the transfer of control from Physician A to Physician B deteriorated the safety control structure. If Physician A would have stayed in charge, she would have likely detected the fact that the lab result shown on sunday were the ones that she already reacted to. Another temporal factor playing a role in this accident is the fact that the shift occurred over the weekend and the daily blood tests were suspended for sunday. Otherwise the overdose would have likely been caught 24 hrs earlier.

Step 9 - ***Generate Recommendations.*** Leveson describes a canonical structure for describing recommendations generated with CAST [20]. Using this structure for presenting the findings generated earlier would be beyond the space constraints of this paper. However, it is clear to see how the above findings would feed into recommended additional safety constraints (*1,...,*13 from Step 6 above), recommended additional communication and coordination links (such as +1,..,+3 above), recommended additional control components (e.g., implementing constraint *5 would imply the addition of a component to watch for alerts indicating EMR user confusion and initiating user training), and other types of changes to the safety control structure.

Compared with the recommendations generated by the hospital committee that investigated the accident, it is clear that the application of STAMP/EMR and CAST provides a more systematic, comprehensive explanation of the causes for the accident and how to improve system safety. The original change recommendations documented in Horsky's report are summarized below:

1. Screens for ordering IV injections and drips fluids need to be clearly distinct
2. Screens that list active medication orders also should list drip orders.
3. Indicate when the most recent lab results are not from the current day.
4. Add an alert that would inform users, ordering potassium (drip or bolus) when the patient already has another active order for potassium.
5. Add an alert informing users ordering potassium when there has not been a serum potassium value recorded in the past 12 hours or the most recent potassium value is greater than 4.0.
6. Make other minor changes to increase the consistency of ordering screens.

The application of STAMP/EMR and CAST yields recommendations that cover all these original recommendations, albeit not at the level concrete medical guidelines (see nr. 5 above). However, STAMP also uncovers important other safety flaws, such as the missing feedback (step 7, +1 above) between physicians ordering medication and nurses administering orders. Without this feedback an overdosing error is likely to reoccur under similar circumstances even if the originally recommended changes are implemented, since a physician cancelling an IV bolus injection order and replacing it with a drip fluid order misses the feedback on whether the cancelled injection was already administered or not.

5 Conclusion

While EMRs have assumed an increasingly critical role for the delivery of modern health care services, there have been growing concerns about the quality of EMR software products, in particular with respect to safety. Many jurisdictions have begun to react to calls for regulating the health software industry, often by extending existing regimes for certifying medical devices to health information system software. However, EMR safety is a complex function of diverse socio-technical-legal factors and current strategies for regulating medical devices do not sufficiently consider these characteristics. In particular, the current focus on pre-market safety evaluations of software products and certification of vendor manufacturing processes does not appear well vested for creating assurances on EMR system safety. Moreover, traditional methods for safety engineering based on linear event chains have only limited applicability in the context of EMR systems. A system-theoretic approach to safety assurance, using the paradigm of continuous control loops enforcing safety constraints at different hierarchical levels (operational, developmental, regulatory, governmental) can provide a more suitable foundation for safety engineering methods in this domain. As a consequence of adopting such a model, regulatory certification programs for EMRs will have to shift their focus to increase the effectiveness of post-market surveillance and controls. The development of models and systematic processes to guide such a notion of continuous monitoring and control of health information systems is an important current research challenge for the software engineering community.

In this paper, we have taken a first step in the direction of providing a concrete method for EMR safety engineering. We presented an adaptation of Leveson's system-theoretic model for safety engineering (STAMP) to the domain of EMRs and showed its application for analyzing the reasons behind a real-world accident involving a serious medication dose error. However, Leveson has demonstrated applicability of STAMP beyond the analysis of accidents, e.g., for designing new systems and monitoring and evolving the performance of existing ones [20]. More research is required to develop and evaluate concrete methods, tools and processes to enable regulators, manufacturers and operators to apply a system-theoretic paradigm to providing safer EMR systems.

References

1. Standard Practice for System Safety MIL-STD-882E. U.S. Dept. of Defense (2005)
2. Classification of medical devices class i or class ii patient management software. Health Canada, File number: 10-111648-902, May 21 (2009)
3. Ammenwerth, E., Shaw, N.: Bad health informatics can kill–is evaluation the answer. Methods Inf. Med. 44(1), 1–3 (2005)
4. Berger, R., Kichak, J.: Computerized physician order entry: Helpful or harmful? Journal of the American Medical Informatics Association 11(2), 100–103 (2004)
5. Birtwhistle, R., Keshavjee, K., Lambert-Lanning, A., Godwin, M., Greiver, M., Manca, D., Lagace, C.: Building a pan-Canadian primary care sentinel surveillance network: initial development and moving forward. JAMIA 22(4), 412 (2009)
6. Brown, M., Shaw, N., Grimm, N., Muttitt, S., Gebran, J.: Electronic health records and patient safety: What lessons can canada learn from the experience of others? Healthcare Quarterly 11(1), 112–119 (2008)
7. Botti, M., Bucknall, T.: Medical error and error in health care. In: Encyclopedia of Medical Decision Making. SAGE Publications (2009)
8. Chang, A., Schyve, P., Croteau, R., O'Leary, D., Loeb, J.: The jcaho patient safety event taxonomy: a standardized terminology and classification schema for near misses and adverse events. Intl. J. for Quality in Health Care 17(2), 95 (2005)
9. FDA. Federal register, vol. 76(31), February 15, rules and regulations. 880.6310 - Medical Data Device Rule (2011)
10. Fried, B.M., Zuckerman, J.M.: Fda regulation of medical software. Journal of Health Law 33(1), 129–140 (2000)
11. Holden, R.: Cognitive performance-altering effects of electronic medical records: an application of the human factors paradigm for patient safety. In: Cognition, Technology & Work, pp. 1–19 (2010)
12. Horsky, J., Kuperman, G., Patel, V.: Comprehensive analysis of a medication dosing error related to cpoe. JAMIA 12(4), 377 (2005)
13. Isaac, T., Weissman, J., Davis, R., Massagli, M., Cyrulik, A., Sands, D., Weingart, S.: Overrides of medication alerts in ambulatory care. Archives of Internal Medicine 169(3), 305 (2009)
14. Karsh, B., Holden, R., Alper, S., Or, C.: A human factors engineering paradigm for patient safety: designing to support the performance of the healthcare professional. Quality and Safety in Health Care 15(suppl. 1), i59 (2006)

15. Kohn, L., Corrigan, J., Donaldson, M., McKay, T., Pike, K.: To err is human. National Academy Press (2000)
16. Koppel, R., Kreda, D.: Health care information technology vendors'" hold harmless" clause: implications for patients and clinicians. JAMA 301(12), 1276 (2009)
17. Koppel, R., Metlay, J., Cohen, A., Abaluck, B., et al.: Role of computerized physician order entry systems in facilitating medication errors. JAMA 293(10), 1197 (2005)
18. Leonard, M., Graham, S., Bonacum, D.: The human factor: the critical importance of effective teamwork and communication in providing safe care. Quality and Safety in Health Care 13(suppl. 1), i85 (2004)
19. Leveson, N.: A new accident model for engineering safer systems. Safety Science 42(4), 237–270 (2004)
20. Leveson, N.: Engineering a Safer World - Systems Thinking Applied to Safety. MIT Press (2011)
21. Leveson, N.G.: Software safety: why, what, and how. ACM Comput. Surv. 18, 125–163 (1986)
22. Parker, J., Coiera, E.: Improving clinical communication. JAMIA 7(5), 453 (2000)
23. Phillips, W., Gong, Y.: Developing a Nomenclature for EMR Errors. In: Jacko, J.A. (ed.) HCI International 2009. LNCS, vol. 5613, pp. 587–596. Springer, Heidelberg (2009)
24. Poissant, L., Pereira, J., Tamblyn, R., Kawasumi, Y.: The impact of electronic health records on time efficiency of physicians and nurses: a systematic review. JAMIA 12(5), 505–516 (2005)
25. Reason, J.: Human error: models and management. Bmj 320(7237), 768 (2000)
26. Runciman, W., Helps, S., Sexton, E., Malpass, A.: A classification for incidents and accidents in the health-care system. J. Qual. in Clin. Practice 18(3), 199 (1998)
27. Schaub, G.H., Hoppner, J.: Software and medical devices - recommendation nb-med/2.2/rec4. European Co-ordination of Notified Medical Devices (NB-MED)
28. Shuren, J.: Testimony of jeffrey shuren. director of fda's center for devices and radiological health information technology (hit) policy committee. adoption/certification workgroup, February 25 (2010
29. Smith, M.: The electronic patient record and computer safety. In: Proceedings Strategic Alliances between Patient Documentation and Medical Informatics, AM-ICE 1995, Rotterdam, Netherlands, pp. 325–331 (1995)
30. Weber-Jahnke, J.H.: A preliminary study of apparent causes and outcomes of reported failures with patient management software. In: 3rd Intl. Workshop on Software Engineering in Health Care (SEHC). ACM Press (2011)
31. Williams, J., Weber-Jahnke, J.H.: Medical device regulation and patient management software. Health Law Journal 18(1) (2010)
32. Zhang, J., Patel, V., Johnson, T., Shortliffe, E.: A cognitive taxonomy of medical errors. Journal of Biomedical Informatics 37(3), 193–204 (2004)

Challenges in eHealth: From Enabling to Enforcing Privacy

Naipeng Dong*, Hugo Jonker, and Jun Pang

Faculty of Sciences, Technology and Communication,
University of Luxembourg, Luxembourg

Abstract. Privacy is recognised as a fundamental requirement for eHealth systems. Proposals to achieve privacy have been put forth in literature, most of which approach patient privacy as either an access control or an authentication problem. In this paper, we investigate privacy in eHealth as a communication problem, since future eHealth systems will be highly distributed and require interoperability of many sub-systems. In addition, we research privacy needs for others than patients. In our study, we identify two key privacy challenges in eHealth: *enforced privacy* and *privacy in the presence of others*. We believe that these privacy challenges are vital for secure eHealth systems, and more research is needed to understand these challenges. We propose to use formal techniques to understand and define these new privacy notions in a precise and unambiguous manner, and to build an efficient verification framework.

1 Introduction

The inefficiency of traditional paper-based health-care and the development of information communication technologies, in particular cloud computing, mobile, and satellite communications, give electronic health-care (eHealth for short) a great opportunity to grow as an important part of people's daily life. eHealth systems aim to provide effective support for secure sharing of information and resources across different health-care settings, and workflows among different health-care providers. The services of such systems for the general public are intended to be more secure, more effective, more efficient, more patient-centered and more timely. However, the attractive advantages of eHealth systems entail many scientific challenges. One of the foremost of these are the privacy issues raised by adapting electronic storage and communication, due to the sensitive nature of health data. Indeed, privacy in eHealth has been recognised as one of the paramount requirements necessary for adoption by the general public [1,2]. Moreover, existing privacy experience from domains such as electronic voting (e.g., [3]) and online auctions (e.g., [4]) does not carry over straightforwardly. In voting and auctions, there is a natural division into two types of roles: participants (voters, bidders) and authorities (who run the election/auction). eHealth systems have to deal with a far more complex constellation of roles: doctors,

* N. Dong was supported by the grant No. PHD-09-027 from FNR Luxembourg.

patients, pharmacists, insurance companies, medical administration, etc. Each of these roles has access to different private information and different privacy concerns. As existing privacy approaches from other domains are not properly equipped to handle such a diverse array of roles, privacy must be tailored to the health-care domain.

Various proposals to achieve privacy in eHealth have been put forth in the literature. Most focus on patient privacy (notable exceptions: [5,6,7]) and approach that as an access control or authentication problem. The virtually exclusive focus on patient privacy in the literature has left the question of privacy requirements of other parties, doctors in particular, wide open. The fact that some parties, e.g., pharmacists, cannot be fully trusted adversely impacts privacy [6].

Furthermore, eHealth systems must provide assurance of privacy and address privacy issues at different system levels – architectural design, access control, communication protocols, etc. Making the issue more complex is the sensitive nature of health-care data, which is subject to regulations, guidelines and application of professional ethics. Currently, (patient) privacy is usually described in terms of protection of information and in terms of controlling access to services. Thus, it is commonly achieved in practice by means of a form of access control or authentication (e.g., see [8,9,10,11,12]). However, typical eHealth systems, especially in future, will be highly distributed and require interoperability of many sub-systems. Even if health-care data is well protected and access control is perfectly employed, improperly designed communication protocols for such interoperability will cause information leakage and hence breach users' privacy. So far, security and privacy of communication protocols in eHealth systems is seldom studied in the literature. This implies that related privacy notions, emerging in other fields, do not yet have a counterpart in the eHealth domain. A prime example is the notion of coercing vs. enforced privacy: in voting, a voter is coerced to reveal how she voted (to sell her vote). For eHealth systems, a similar case would be a pharmaceutical company bribing a doctor to reveal which medicine he prescribed. Given the current state of eHealth research, we find that *enabling* privacy is well-studied. However, there is a lack of attention for *enforcing* privacy.

Contributions. Our main contribution is to identify two new types of enforcing privacy as key privacy challenges for the field: *enforced privacy* (e.g., a doctor cannot prove to a pharmaceutical company which medicine he prescribed) and *privacy in the presence of others* (e.g., a patient cannot reveal which doctor prescribed her medicine). We propose to use formal techniques to address these challenges, that is: to understand and interpret these new privacy notions in a precise and unambiguous manner, and to build an efficient verification framework for analysing privacy properties of eHealth systems.

Outline of the Paper. We briefly survey existing approaches to privacy in eHealth in Section 2, finding a lack of attention to enforced privacy. Next, Section 3 identifies what we argue to be two main challenges for privacy in eHealth. In the remainder of the paper, we outline the need for a formal approach in addressing these challenges (Section 4), and report briefly on our ongoing analysis of an eHealth protocol claiming to be privacy-preserving (Section 5).

2 A Brief Survey of Privacy in eHealth

Privacy in eHealth systems has attracted much research effort and a variety of different privacy-enabling methods have been proposed. This section provides a brief overview[1] of previous work on privacy in eHealth. We divide previous work in two categories, focusing on patient privacy (Section 2.1) and on doctor privacy (Section 2.2), respectively.

2.1 Enabling Patient Privacy

The importance of patient privacy in eHealth is traditionally seen as vital to establishing a good doctor-patient relationship. This is even more pertinent with the emergence of the Electronic Patient Record [8]. As in most of the literature, a necessary early stage of eHealth is to transform the paper-based health-care process into a digital process. The most important changes in this stage are made to patient information processing, mainly health-care records. To properly express privacy requirements for such patient records, privacy policies are considered the de facto standard. There are three main approaches to implement these requirements of patient privacy: access control, architectural design, and the use of cryptography.

Patient Privacy by Access Control. To preserve privacy of electronic health-care records, one necessary part is to limit access to these records to allowed parties. Anderson [8] lists several privacy threats to personal health information as support for his claim that privacy policies for access control should be mandatory in eHealth systems. Anderson [8] proposes access rules to restrict the number of users who can access any record and the maximum number of records accessed by any user. In a case study [9], Louwerse argues that consent-based access rules (e.g., a patient approving use of his data for research) are required in addition to rules based on the "need-to-know" principle. Evered and Bögeholz [11] investigate minimal-disclosure access constraints for a small eHealth system, and find that even for a small system, constraints cannot be expressed easily or clearly using static access rules. As a solution, they propose adding a middle layer of logic, which translates constraints into access rules. Reid et al. [10] adapt role-based access control (RBAC) to include explicit consent and denial. Explicit denial is to grant access to a role (e.g., doctors), but deny access to a particular individual (excluding a particular doctor); explicit consent is the converse property: granting access for individuals while denying access to the role. Kalam et al. claim also investigate shortcomings of classical access control models (such as RBAC and task based authorisation controls (TBAC) [13], etc.), and find these are insufficient to capture security policies that need to be context-aware (e.g., to grant emergency access to a patient record), that specify onligations or recommendations. They propose a new access control model OrBAC [14] (organisation based access control), designed to be particularly suited for eHealth access control.

[1] This literature overview is not intended to be exhaustive.

The access rules of eHealth systems are complex, and may become inconsistent – one rule may contradict another. Cuppens et al. [15] propose assigning priorities to rules and show that this can resolve such problems in rule-based access control and in OrBAC.

Patient Privacy by Architectural Design. As stated in the introduction, eHealth systems cater to a number of different roles, including doctors, patients, pharmacists, insurers, etc. Each such role has its own sub-systems or components. As such, eHealth systems can be considered as a large network of systems, including administrative system components, laboratory information systems, radiology information systems, pharmacy information systems, and financial management systems. Diligent architectural design is an essential step to make such a complex system function correctly. Since privacy is important in eHealth systems, keeping privacy in mind when designing the architecture of such systems is a promising path towards ensuring privacy [16]. Ko et al. [17] discuss privacy issues when building wireless sensor networks for eHealth. Some eHealth system architectures are specially designed with a particular privacy issue in mind, e.g., Maglogiannis et al. [18] propose an architecture that enhances patient location privacy by communication via proxies, which can learn location but not patient identity. There also exist architectures which use different privacy protecting techniques at different layers of a system. For example, Chiu et al. [19] study privacy requirements for cross-institution image protection and design a system that uses access control rules, RBAC, and watermarking at various levels to offer secure and privacy-aware, cross-institutional image sharing.

Cryptographic Approaches to Patient Privacy. Cryptography is a necessary tool for privacy in eHealth systems [20]. For example, Van der Haak et al. [21] use digital signatures and public-key authentication (for access control) to satisfy legal requirements for cross-institutional exchange of electronic patient records. Ateniese et al. [22] use pseudonyms to preserve patient anonymity, and enable a user to transform statements concerning one of his pseudonyms into statements concerning one of his other pseudonyms (e.g., transforming a prescription for the pseudonym used with his doctor to a prescription for the pseudonym used with the pharmacy). Layouni et al. [23] consider communication between health monitoring equipment at a patient's home and the health-care center. They propose a protocol using wallet-based credentials (a cryptographic primitive) to let patients control when and how much identifying information is revealed by the monitoring equipment. More recently, De Decker et al. [7] propose a health-care system for communication between insurance companies and administrative bodies as well as patients, doctors and pharmacists. Their system relies on various cryptographic primitives to ensure privacy, including zero-knowledge proofs, signed proofs of knowledge (a signature scheme which uses zero-knowledge proofs to sign a message), and bit-commitments. Their system is explained in more detail in Section 5.

2.2 Ensuring Doctor Privacy

A relatively understudied privacy aspect is that of doctor privacy. Matyáš [5] investigates the problem of enabling analysis of prescription information while ensuring doctor privacy. His approach is to group doctors, and release the data per group, hiding who is in the group. He does not motivate a need for doctor privacy, however. Two primary reasons for doctor privacy have been identified in the literature: (1) (Ateniese et al. [22]) to safeguard doctors against administrators setting specific efficiency metrics on their performance (e.g., requiring the cheapest medicine be used, irrespective of the patient's needs). To address this, Ateniese et al. [6,22] propose an anyonymous prescription system that uses group signatures for privacy; (2) (De Decker et al. [7]) to prevent a pharmaceutical company from bribing a doctor to prescribe their medicine. De Decker et al. also note that preserving doctor privacy is not sufficient to prevent bribery: pharmacists could act as *go-betweens*, revealing the doctor's identity to the briber. They propose a privacy-preserving health-care scheme that incorporates the roles of pharmacist and health insurer as well as doctor and patient.

2.3 Observations

In the above overview, we observe that current approaches to privacy in eHealth mostly focus on patient privacy and try to solve it as an access control or authentication problem. However, eHealth systems involve many different roles, and these roles have their own privacy concerns. We believe that doctor privacy is as important as patient privacy and should be studied in more depth to avoid situations such as doctor bribery (cf. Section 2.2). In addition, we consider that one party's privacy may depend on another party (e.g., in the case of a pharmacist revealing prescription behaviour of a doctor). Our opinion is that offering privacy is insufficient if privacy can be reduced in such ways.

It is clear from the analysis that privacy in eHealth systems needs to be addressed at different layers: use of cryptography guarantees privacy at the foundation layer; access control ensures privacy at the service layer; privacy by design addresses privacy concerns at the system/architecture layer. Since eHealth systems are complex [24] and rely on correct communications between many sub-systems, we strongly advocate to study privacy in eHealth as a communication problem. In fact, message exchanges in communication protocols may leak information which leads to a privacy breach.

Privacy properties such as anonymity, unlinkability, untraceability etc. have been studied in the literature. All these notions play a role in eHealth systems and each provides a different strength of privacy that can be *enabled*. However, enabling privacy is far from enough. In many cases, a system must *enforce* user privacy instead of allowing the user to pursue it. Enforced privacy has been considered in other domains. Below, we briefly sketch highlights in development of the notion of enforced privacy from other domains.

Enforced Privacy in Other Domains. In the literature, the notion of enforced privacy was studied first in electronic voting. Benaloh and Tuinstra [25] introduce the notion of *receipt-freeness*, which expresses that a voter cannot gain any information to prove to a vote-buyer how she voted. This notion preserves voter-privacy even when a voter actively seeks to renounce that privacy, as in the case of vote-buying. Another, stronger notion of privacy is coercion-resistance [26], stating that a voter cannot cooperate with the intruder to prove how she voted. Both notions of privacy actually capture the essential idea that privacy must be enforced by a system upon its users, instead of merely offering it.

Enforced privacy has been studied outside voting. For instance, a few papers [27,28] have identified a need for receipt-freeness in online auctions. In eHealth, however, enforced privacy has received to date little attention.

3 Key Privacy Challenges

Considering how the notion of enforced privacy applies to the eHealth domain leads us to identify two key privacy challenges for the domain:

– *enforced privacy*, e.g., a doctor cannot prove to a pharmaceutical company which medicine he prescribed; and
– *privacy in the presence of others*, e.g., a patient cannot help a doctor to prove he prescribed her medicine.

Satisfying these privacy notions is not easily in any setting. However, the added complexity of the eHealth domain (where a "break-the-glass" requirement exists to ensure emergency access to records) makes these formidable challenges.

3.1 Challenge I: Enforced Privacy

Enforced privacy plays an important role in eHealth systems, especially for doctors. A typical scenario can be described as follows. A pharmaceutical company seeks to persuade a doctor to favor a certain kind of medicine by bribing or coercing. To prevent this, a doctor should not be able to prove which medicine he is prescribing to this company (in general, to an adversary). This implies that doctor privacy must be enforced by eHealth systems. Generally speaking, a doctor should not be able to prove what he prescribed to any third party except for trusted authorities.

Enforced privacy in eHealth is hardly studied. As such, a proper understanding (beyond the anecdotal scenario given above) of the importance of enforced privacy is absent. Therefore, it is important to investigate which roles in eHealth systems require the notion of enforced privacy. It is also interesting to study which cryptographic techniques can be employed to enforce privacy. Development of systems providing enforced privacy will benefit from privacy-enforcing techniques used in other domains. These include techniques to guarantee receipt-freeness and coercion-resistance, for example chameleon bit commitments as used in a receipt-free online auction protocol [27].

To tackle this challenge, we propose to first focus on lifting formal definitions of enforced privacy to the eHealth domain, and secondly to develop efficient techniques to verify enforced privacy in eHealth. Due to the complexity of eHealth systems, all these remain as scientific challenges.

3.2 Challenge II: Privacy in the Presence of Others

The notion of enforced privacy emerged in voting systems and online auction protocols. In these domains, privacy requirements are mainly focused on the central role (voter and bidder, respectively). In stark contrast, eHealth systems involve many different, non-central roles. Some of these roles have access to sensitive information which reveals something about the privacy of another role. For example, a pharmacist has access to prescriptions, and thus knows something about the prescription behaviour of a doctor. Such a party may be bribed or coerced to help break the other party's privacy. Literature [5,7,22] underlines the need to protect a doctor's prescription pattern. This means that no one, except for the doctor himself or trusted third parties, must be able to link the doctor to his prescriptions. In order to obtain a doctor's prescription pattern, an adversary can bribe other parties to reveal their private information which lets the adversary determine a doctor's prescription. This leads us to formulate the requirement of *third-party-independent doctor privacy*: no third party should be able to help the adversary link a doctor to his prescription. On the other hand, these third parties can also help to protect a coerced user. We therefore distinguish two cases:

– *coalition-enforced privacy* (CE-PRIV): a third party helping to protect the coerced user's privacy; and
– *third-party-independent privacy* (TP-INDEP): a third party helping the the adversary to break a user's privacy.

In the first case, the third party cooperates with the coerced user to protect the coerced user's privacy, and reveals his secret information to the coerced user if necessary – forming a coalition, which enables the coerced user to hide from the adversary. For example, a patient cooperates with a bribed doctor to lie about which medicine the doctor prescribed. In the second case, the third party reveals (whether by choice or coercion) his private information to the adversary, enabling the adversary to breach the other user's privacy. For example, a pharmacist can help to breach doctor privacy, by revealing the doctor's prescription behaviour to a pharmaceutical company. In some cases (such as the example), the doctor's privacy is breached without involving the doctor.

We emphasise that TP-INDEP is different from enforced privacy: in enforced privacy, the revealing party breaches her own privacy, while in TP-INDEP, she helps breach another's privacy.

CE-PRIV and TP-INDEP are new notions of privacy, which have not been studied in the literature. In our view, these privacy notions are important for eHealth systems, and stand on their own as privacy challenges.

4 The Need for a Formal Approach

We believe that to solve the two key challenges in a generic fashion, an improved understanding of the concepts *enforced privacy* and *privacy in the presence of others* is necessary. Moreover, we feel that an evaluation method is necessary to validate that a proposed solution indeed addresses these challenges. We argue that neither understanding nor evaluation framework can be properly addressed without formal methods.

In the literature, many research efforts have been devoted to ensure enforced privacy properties for electronic voting. However, despite the best intentions (e.g., [25,29]), receipts have time and again been found (e.g., [30,31]). This propose-attack cycle underlines the need for formal methods, which are mathematically based techniques to specify and verify systems.

This is especially pertinent in the eHealth domain, where enforced privacy is a new concept. Moreover, in the eHealth domain, privacy has focused on access control and system design – but even under the assumption of perfect cryptography, communications may reveal private information (cf. the above mentioned attacks). Finding such attacks manually is error-prone, and can give no assurances of completeness. On the other hand, formal verification can give some assurances. Since the eHealth domain stands to benefit so strongly from employing formal methods to express and evaluate security requirements, we fiercely advocate its use.

Current Formal Approaches to Enforced Privacy. In voting, several formalisations of enforced privacy properties have been proposed. Delaune et al. [3] develop their formalisation of receipt-freeness and coercion-resistance based on observational equivalences in the applied pi calculus [32]. Automatic verification techniques within the applied pi calculus framework have been proposed by Backes et al. [33]. Their approach focuses on remote electronic voting protocols, and mainly deals with coercion-resistance. Baskar et al. [34] define a language to specify voting protocols and use an epistemic logic to reason about receipt-freeness. Although it is relatively easy to express privacy properties based on logic of knowledge, it is rather difficult to develop verification techniques within such a logical framework. Jonker et al. [35] introduce a formal framework combining knowledge reasoning and trace equivalences to model and reason about receipt-freeness for voting protocols and provided a quantitative definition of voter-controlled privacy. Based on the work of Delaune et al. [3], Dong et al. [4] formalise receipt-freeness in online auction.

The use of process theory has led to success in voting, and the possibility of lifting this to other domains has been shown. However, an eHealth system has a large diversity in roles, something not seen in voting systems or auction systems. As such, a direct applications of formalisms developed elsewhere to eHealth can only approximate the subtleties of (coalition-)enforced privacy in eHealth. It is still a challenge to formally define and verify privacy, enforced privacy, privacy in the presence of other in eHealth.

5 Towards Formalising Enforced Privacy

A Case Study. Currently, we are investigating[2] privacy of the DLV08 protocol [7]. The protocol involves five main roles: patient, doctor, pharmacist, medical prescription administration (MPA), and insurance company. It works as follows: first, a patient communicates with the doctor to get a prescription. The patient then communicates with the pharmacist to receive the medication prescribed. Next, the pharmacist communicates with the MPA for a refund. The MPA sends an invoice to the patient's insurance company, and once this is paid, the MPA pays the pharmacist.

We model each role's behaviour as a process in applied pi. Thus, the protocol is modelled as the parallelisation of all these processes[3]: $P_{dr} \mid P_{pt} \mid P_{ph} \mid P_{mpa} \mid P_{hii}$. The DLV08 protocol claims (amongst others) doctor-enforced privacy (a doctor cannot prove what he prescribed), and third-party-enforced doctor privacy (third parties cannot help the adversary to reveal the doctor's prescription pattern). To check these claims, we formalise doctor-enforced privacy and third-party-independent doctor privacy as observational equivalences and verify whether DLV08 satisfies either using the automatic verification tool ProVerif [36].

To illustrate how we approach the formalisation, we sketch our formalisation of doctor-enforced privacy. Intuitively, doctor-enforced privacy means that the adversary cannot distinguish between a doctor prescribing a and claiming to have prescribed a, and a doctor prescribing b while claiming to have prescribed a. We model this roughly as $P_{dr}(a, a) \approx P_{dr}'(b, a)$, where $\approx$ denotes observational equivalence. Note that the doctor is lying in the second case, thus he behaves differently – hence we do not write $P_{dr}(b, a)$. Naturally, we do not check this in isolation, but in the presence of all other parties, i.e., we verify whether $P_{dr}(a, a) \mid P_{pt} \mid P_{ph} \mid P_{mpa} \mid P_{hii} \approx P_{dr}'(b, a) \mid P_{pt} \mid P_{ph} \mid P_{mpa} \mid P_{hii}$.

Future Directions. Existing formalisations (in voting) of enforced privacy using observational equivalence [3] provide voters with a fixed defensive strategy. This approach implies that the coerced voter is teamed up with another voter, such that one of their two cast votes matches the adversary's wishes. Privacy is preserved if the adversary cannot determine which of them cast his vote. This forming of defensive coalitions[4] was introduced as a modelling trick to ensure observational equivalence between an execution where a coerced voter complies with the coercer and one where she does not. In general, we envisage more applications for coalition-forming in the formal understanding of enforced privacy. On one hand, different parties may form larger coalitions and so have a more robust defensive model. On the other hand, as noted in Challenge II, privacy with respect to an adversary conspiring with multiple parties is inherently lower than the privacy with respect to an adversary without such a coalition or with a smaller coalition. This leads to a variety in behaviour, which is not easily expressed in process theory, but naturally captured in game theory. Hence, game

[2] For the latest developments, see `http://satoss.uni.lu/projects/epriv/`

[3] We omit some details, such as multiple instances of processes, here.

[4] Receipt-freeness and coercion-resistance in [3] match our notion of CE-PRIV.

theoretic approaches towards enforced privacy may be promising. In particular, the work of Küsters et al. [37], a game theoretic definition of coercion-resistance in voting, might be adapted towards this end.

Assurance of privacy properties via formal verification is an important step in developing eHealth systems. However, automatic verification of observational equivalences is in general undecidable[5] . Recently, research has been devoted to decision procedures for observational equivalences by focusing on a fragment of the applied pi calculus [38]. It is interesting to investigate the applicability of this research to aid verification of privacy properties in eHealth.

6 Conclusion

eHealth systems are drawing attention because of their potential advantages. However, due to several challenges, the widespread adoption of eHealth systems is still at an early stage. One of the key challenges is to understand privacy issues in eHealth. Current study on this topic mainly focuses on patient privacy and solves it as an access control problem. Privacy issues of other involved parties and during communications are rarely studied so far. We advocated the position that in addition to *enabling* privacy in eHealth (such as protecting patient privacy), *enforcing* privacy plays a critical role, especially for doctors. In addition, we identified another privacy requirement, *privacy in the presence of others*, and argued that a proper understanding requires formalisation, as does genuine verification of these properties. Finally, we sketched our ongoing study of the DLV08 protocol, which claims to enforce privacy for the doctor.

References

1. Meingast, M., Roosta, T., Sastry, S.S.: Security and privacy issues with health care information technology. In: Proc. 28th Annual Conference of the IEEE Engineering in Medicine and Biology Society, pp. 5453–5458. IEEE CS (2006)
2. Kotz, D., Avancha, S., Baxi, A.: A privacy framework for mobile health and home-care systems. In: Proc. Workshop on Security and Privacy in Medical and Home-Care Systems, pp. 1–12. ACM Press (2009)
3. Delaune, S., Kremer, S., Ryan, M.D.: Verifying privacy-type properties of electronic voting protocols. Journal of Computer Security 17, 435–487 (2009)
4. Dong, N., Jonker, H.L., Pang, J.: Analysis of a Receipt-Free Auction Protocol in the Applied Pi Calculus. In: Degano, P., Etalle, S., Guttman, J. (eds.) FAST 2010. LNCS, vol. 6561, pp. 223–238. Springer, Heidelberg (2011)
5. Matyáš, V.: Protecting doctors' identity in drug prescription analysis. Health Informatics Journal, 205–209 (1998)
6. Ateniese, G., de Medeiros, B.: Anonymous e-prescriptions. In: Proc. ACM Workshop on Privacy in the Electronic Society, pp. 19–31. ACM Press (2002)
7. De Decker, B., Layouni, M., Vangheluwe, H., Verslype, K.: A Privacy-Preserving eHealth Protocol Compliant with the Belgian Healthcare System. In: Mjølsnes, S.F., Mauw, S., Katsikas, S.K. (eds.) EuroPKI 2008. LNCS, vol. 5057, pp. 118–133. Springer, Heidelberg (2008)

[5] ProVerif provides limited support for a specific class of protocols.

8. Anderson, R.: A security policy model for clinical information systems. In: Proc. 17th IEEE Symposium on Security and Privacy, pp. 30–43. IEEE CS (1996)
9. Louwerse, K.: The electronic patient record; the management of access – case study: Leiden University hospital. International Journal of Medical Informatics 49, 39–44 (1998)
10. Reid, J., Cheong, I., Henricksen, M., Smith, J.: A Novel Use of rBAC to Protect Privacy in Distributed Health Care Information Systems. In: Safavi-Naini, R., Seberry, J. (eds.) ACISP 2003. LNCS, vol. 2727, pp. 403–415. Springer, Heidelberg (2003)
11. Evered, M., Bögeholz, S.: A case study in access control requirements for a health information system. In: Proc. 2nd Australian Information Security Workshop. Conferences in Research and Practice in Information Technology, vol. 32, pp. 53–61. Australian Computer Society (2004)
12. Hung, P.C.K.: Towards a privacy access control model for e-healthcare services. In: Proc. 3rd Annual Conference on Privacy, Security and Trust (2005)
13. Thomas, R.K., Sandhu, R.S.: Task-based authorization controls (TBAC): A family of models for active and enterprise-oriented authorization management. In: Proc. 11th Conference on Database Security. IFIP Conference Proceedings, vol. 113, pp. 166–181. Springer (1997)
14. Kalam, A., Benferhat, S., Miège, A., Baida, R., Cuppens, F., Saurel, C., Balbiani, P., Deswarte, Y., Trouessin, G.: Organization based access control. In: Proc. 4th IEEE Workshop on Policies for Distributed Systems and Networks, pp. 120–131. IEEE CS (2003)
15. Cuppens, F., Cuppens-Boulahia, N., Ghorbel, M.B.: High level conflict management strategies in advanced access control models. Electronic Notes in Theoretical Computer Science 186, 3–26 (2007)
16. Sneha, S., Varshney, U.: Enabling ubiquitous patient monitoring: Model, decision protocols, opportunities and challenges. Decision Support Systems 46, 606–619 (2009)
17. Ko, J., Lu, C., Srivastava, M.B., Stankovic, J.A., Terzis, A., Welsh, M.: Wireless sensor networks for healthcare. Proceedings of IEEE 98, 1947–1960 (2010)
18. Maglogiannis, I., Kazatzopoulos, L., Delakouridis, C., Hadjiefthymiades, S.: Enabling location privacy and medical data encryption in patient telemonitoring systems. IEEE Transactions on Information Technology in Biomedicine 13, 946–954 (2009)
19. Chiu, D.K.W., Hung, P.C.K., Cheng, V.S.Y., Kafeza, E.: Protecting the exchange of medical images in healthcare process integration with web services. In: Proc. 40th Hawaii Conference on Systems Science, pp. 131–140. IEEE CS (2007)
20. Biskup, J., Bleumer, G.: Cryptographic protection of health information: cost and benefit. International Journal of Bio-Medical Computing 43, 61–67 (1996)
21. van der Haak, M., Wolff, A.C., Brandner, R., Drings, P., Wannenmacher, M., Wetter, T.: Data security and protection in cross-institutional electronic patient records. International Journal of Medical Informatics 70, 117–130 (2003)
22. Ateniese, G., Curtmola, R., de Medeiros, B., Davis, D.: Medical Information Privacy Assurance: Cryptographic and System Aspects. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, pp. 199–218. Springer, Heidelberg (2003)
23. Layouni, M., Verslype, K., Sandıkkaya, M.T., De Decker, B., Vangheluwe, H.: Privacy-Preserving Telemonitoring for eHealth. In: Gudes, E., Vaidya, J. (eds.) Data and Applications Security 2009. LNCS, vol. 5645, pp. 95–110. Springer, Heidelberg (2009)

24. Tien, J.M., Goldschmidt-Clermont, P.: Healthcare: A complex service system. Journal of Systems Science and Systems Engineering 18, 257–282 (2009)
25. Benaloh, J., Tuinstra, D.: Receipt-free secret-ballot elections (extended abstract). In: Proc. 26th Symposium on Theory of Computing, pp. 544–553. ACM Press (1994)
26. Juels, A., Catalano, D., Jakobsson, M.: Coercion-resistant electronic elections. In: Proc. 4th ACM Workshop on Privacy in the Electronic Society, pp. 61–70. ACM Press (2005)
27. Abe, M., Suzuki, K.: Receipt-Free Sealed-Bid Auction. In: Chan, A.H., Gligor, V.D. (eds.) ISC 2002. LNCS, vol. 2433, pp. 191–199. Springer, Heidelberg (2002)
28. Chen, X., Lee, B., Kim, K.: Receipt-free Electronic Auction Schemes Using Homomorphic Encryption. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 259–273. Springer, Heidelberg (2004)
29. Lee, B., Kim, K.: Receipt-free electronic voting through collaboration of voter and honest verifier. In: Proc. Japan-Korea Joint Workshop on Information Security and Cryptology, pp. 101–108 (2000)
30. Hirt, M., Sako, K.: Efficient Receipt-Free Voting Based on Homomorphic Encryption. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 539–556. Springer, Heidelberg (2000)
31. Lee, B., Kim, K.: Receipt-free Electronic Voting with a Tamper-resistant Randomizer. In: Deng, R.H., Qing, S., Bao, F., Zhou, J. (eds.) ICICS 2002. LNCS, vol. 2513, pp. 389–406. Springer, Heidelberg (2002)
32. Abadi, M., Fournet, C.: Mobile values, new names, and secure communication. In: Proc. 28th Symposium on Principles of Programming Languages, pp. 104–115. ACM Press (2001)
33. Backes, M., Hriţcu, C., Maffei, M.: Automated verification of remote electronic voting protocols in the applied pi-calculus. In: Proc. 21st IEEE Computer Security Foundations Symposium, pp. 195–209. IEEE CS (2008)
34. Baskar, A., Ramanujam, R., Suresh, S.: Knowledge-based modelling of voting protocols. In: Proc. 11th Conference on Theoretical Aspects of Rationality and Knowledge, pp. 62–71. ACM Press (2007)
35. Jonker, H.L., Mauw, S., Pang, J.: A formal framework for quantifying voter-controlled privacy. Journal of Algorithms in Cognition, Informatics and Logic 64(2-3), 89–105 (2009)
36. Blanchet, B.: An efficient cryptographic protocol verifier based on prolog rules. In: Proc. 14th IEEE Computer Security Foundations Workshop, pp. 82–96. IEEE CS (2001)
37. Küsters, R., Truderung, T., Vogt, A.: A game-based definition of coercion-resistance and its applications. In: Proc. 23rd IEEE Computer Security Foundations Symposium, pp. 122–136. IEEE CS (2010)
38. Cortier, V., Delaune, S.: A method for proving observational equivalence. In: Proc. 22nd IEEE Computer Security Foundations Symposium, pp. 266–276. IEEE CS (2009)

A Technique for Strengthening Weak Passwords in Electronic Medical Record Systems

Samuel Tusubira Kalyango and Gilbert Maiga

School of Computing and Informatics Technology,
Makerere University Box 7062 kampala, Uganda
tusubirask@yahoo.com, gmaiga@cit.mak.ac.ug

Abstract. The internet has accelerated access to and sharing of electronic medical records (EMR). EMRs are meant to be confidential and only accessed or shared with authorization from the owner. A combination of UserID and a Password is the most widely used mechanism to assure user authentication and access to EMRs. However, these mechanisms have been greatly compromised by guessing and hacking of weak passwords leading to increased cases of medical identity theft, cyber terrorism and information systems attacks. This has resulted in false financial claims, debts due to unauthorized disclosure of the private and confidential EMRs leading to huge losses for the victims. This study developed a technique to strengthen weak passwords that integrates UserIDs, weaker password, salts, challenge responses and random variables to derive a stronger password for authentication. A system prototype to test the technique was built, tested and validated by users.

Keywords: Strengthening passwords, Weak passwords, Strengthening weak passwords, Electronic medical records, Strong passwords.

1 Introduction

Online EMRs are a leading cause of the increasing cases of medical identity theft. Password authentication is the most common user identity authentication technique, yet if users don't choose and protect strong passwords all the security efforts are wasted. The choice and use of weak passwords has accelerated medical identity theft [10][2]. Weak passwords give a great opportunity for hackers and attackers of information systems and yet medical identity theft victims have limited rights and resources [21][22][33]. The increasing use of the internet for sharing electronic medical records has led to a corresponding rise in cases of medical identity theft with hackers and cyber terrorists exploiting weak passwords to launch severe system attacks [1][2][19].

Users often choose weaker and easy to remember passwords for several systems and websites. This is as opposed to strong passwords formulated from a combination of letters, numbers, and special alphanumeric characters of at least 8 characters in length, changed over a period of time and not written down. Strong passwords can't be easily guessed by attackers, and yet users have to be able to remember them without referring to written documents [23] [27]. Users tend to forget complex and lengthy passwords especially when not used often. In this paper, we report on the results of a study that was motivated by the need to develop techniques that enable users to devise strong passwords without the burden of having to recall them [17][34].

This paper therefore reports on a technique developed for authenticating system users by strengthening weak passwords in an effort to improve upon the security and safety of electronic medical record systems. The technique integrates UserIDs, weaker password, salts, challenge responses and random variables to derive an architecture for a stronger password authentication mechanism. The developed technique retains the advantages of traditional password authentication mechanisms like low cost of implementation, simplicity of use, user adaptability and system interoperability. Unauthorized access to electronic medical records with the resulting medical identity theft can thus be minimized. The technique was validated by users using a prototype system.

The rest of this paper is structured as follows. Section two presents the need for medical electronic records and explains the concept of medical identity theft. Section three discusses the different types of user authentication mechanisms. In section four, a critique of the current techniques for password creation and strengthening is presented with emphasis on electronic medical records. In section five, the steps for deriving a technique to strengthen weak passwords that integrates the use of challenge responses, a hash algorithm, random variables and salt are presented. The discussion of the findings follows in section six.

2 Electronic Medical Records

Electronic medical records (EMRs) are computer-based health-related information on an individual that can be created, gathered, managed and consulted by authorized clinicians and staff within one health care organization [8] [18]. An EMR system or computer application contains clinical data, provides support for clinical decision making, uses a controlled medical vocabulary, accepts computerized entry of orders by providers for medications and diagnostic tests, and has other features for clinical documentation. Using EMRs, healthcare teams document, monitor, and manage services within a care delivery organization (CDO). The data in the EMR constitutes the legal record of what happened to the patient during their encounter at the CDO, and the EMR is owned by the CDO [11].

EMRs also provide patient information for computerized functions such as drug-to-drug interactions, recommended care practices or interpretation of data to support and improve clinical decisions [18]. They also have health-related information about a patient available from multiple locations and systems. However, if this health related information is presented through a common interface, it improves the ability of clinical personnel to support the best possible diagnosis, treatment, and health management decisions for and with an individual [18]. Electronic medical or health records through employment of ICT have facilitated better capturing and transference of patient information by enhancing patient safety through first of all reducing errors attributed to incorrect or incomplete patient information; secondly by improving access to health care services by streamlining processes, and; thirdly by reducing the cost of delivering care through productivity gains [12]. Computerized decision support systems are effective in enhancing clinical performance for many aspects of health care, including prevention, prescribing of drugs, diagnosis and management, and detection of adverse events and disease outbreaks [5].

EMRs initiatives facilitated by the internet have led to a realization of many benefits to medical practitioners, researchers and patients. These are: i) improved coordination of care within the health care delivery system by increased sharing of health information among all authorized clinicians; ii) providing individuals with electronic access to their own health and wellness information, engaging them in opportunities for improving their health and well-being; and iii) improved community health using aggregated health data for study, public health, emergency preparedness, and quality improvement efforts [6] [9]. EMRs also provide benefits by making clinical notes legible, supporting decision making for ordering drugs, reminding health personnel to stop prescribing drugs and administering vaccines, supporting monitoring programs through provision of reports on outcomes, budgets, and supplies as well as supporting health practitioners in analysis and in managing chronic diseases such as diabetes, hypertension, and heart failure [28].

2.1 Medical Identity Theft

Identity theft, which includes all crimes where someone wrongfully obtains and uses another person's personal data, typically for economic gain, is growing rapidly [10]. One party fraudulently assumes another's identity for economic and personal gain thereby hurting the legitimate person. In the physical world, a person's identity is concrete and must be supported by legal documents, which is different from computing, where even digital data such as passwords, account names, screen names, and logins are used to verify one's identity in order to access services.

Medical identity theft (MIT) is an occurrence when someone uses another person's identity information to make false claims for medical services or goods, resulting in short and long-lasting harm to an individual interacting with the healthcare continuum. MIT has grown and claims more than half a million victims each year yet it still remains difficult to detect as well as repair the damages thereof. Recovery from the damages of MIT theft can take years yet pursuing the criminals is hard [10]. MIT thieves using false access and claims to medical services have led to huge patient financial losses, harassment by debt collectors, and subsequent inability of the victims to find employment [9] [10].

3 User Authentication Mechanisms

User identity authentication basically relies on three options namely: i) where a system user retains physical control of the device e.g. a laptop computer; ii) where a system user presents something he or she knows, such as a password; and iii) something about the system user say biometrics like fingerprint or iris pattern scan [31]. Several user identity authentication mechanisms are in use including biometrics and smart cards. However, issues of user-acceptance, cost of implementation and ease-of-use make the combination of UserID and password authentication as the priority choice in internet applications for the foreseeable future [3][16]. Many information security professionals have focused more on installing complicated security devices like firewalls and intrusion detection systems than on strengthening user passwords. However, passwords remain vital and form a basis for building strong security for information systems [34] [23]. The traditional UserID and password authentication illustrated in Fig. 1 remains the most widely used mechanism for authenticating system users worldwide.

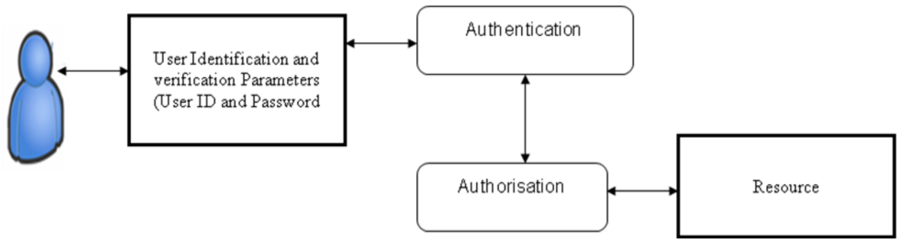

User Identification and verification Parameters (User ID and Password
Authentication
Authorisation
Resource

Fig. 1. High-level Diagram of Traditional UserID and Password Authentication [25]

Simplicity of use and adaptability, cost of implementation and maintenance are some of the factors that still make this mechanism the top priority. On the other hand this mechanism also presents some easier system penetration and information security compromises especially with the weak passwords that system users choose. System users go for weak passwords for ease of remembrance to avoid denial of access to system services if the complex password were to be forgotten. Being the most widely used user authentication mechanism, several security measures have been put in place to counteract the weaknesses presented by the mechanism [16][19] [20].

The SiteKey technique is widely applied in knowledge-based systems especially banking institutions as a user authentication mechanism to deter identity theft. It deploys a multifactor-factor authentication over existing password-based single-factor customer authentication systems. The technique involves a high degree of memorizing the Login ID, password, a selectable image, personal question challenge- response. Some have added a "device ID" to the customer's computer, but if no device ID can be retrieved from the customer's computer, they simply fall back on asking the customer or phisher to supply answers to personal questions. However, his technique has several vulnerabilities e.g. the potential for people to be tricked into divulging logins, passwords, and the answers to personal questions and images being copied and re-used. Also storage and transmission of the secret parameters, say password, is a challenge [22] [33].

The SiteKey places low significance on the challenge questions presenting risk of permanent bypass of challenge questions when a bypass token is present on computer. This leaves the authentication mechanism dependent on only the password which when weak increases the system vulnerability levels. The focus of the SiteKey is not on making the password strong but rather having the challenge question and password. The bypass token is not cryptographically secured since it is simply a large random number which can be regained by man-in-the middle attacks. A system user is also not given chance to revoke privileges to a computer previously permanently connected through a bypass token [14].

In a client server setting, Challenge Response Authentication Mechanisms (CRAM) has been utilized for user authentication. CRAM is a two-level scheme for authenticating network users that is used as part of the Web's Hypertext Transfer Protocol (HTTP). The two levels of CRAM are basic and digest authentication. These present challenges of Man-in-Middle attacks and only enhanced by other methods like Needham-Schroeder based and Kerberos which limit the replay attacks [7][16] [23].

4 Current Techniques for User Password Creation and Strengthening

There are two general approaches to password creation by users. The first general approach is where system users choose their own passwords and store them where they think it is safe. An example is Microsoft's Passport initiative and Password Safe application where users log on and are authenticated on several sites using a single password. The limitation here is when one has to make technical changes on the part of every site that deploys these passports for authentication. In addition, system users are said to be very cautious about placing trust in privacy sensitive information under one centralized database [23]. The second approach is the assignment of fixed passwords to users for accessing several sites or services. An example to this is the Lucent Personal Web Assistant (LPWA) where users access sites or services using a master username and password [20]. In this approach users are provided access using a master username and password derived from a hash-based function of the user's master password and the domain name of the website. Developing techniques for strengthening passwords for system users in order to counteract brute force attacks and hackers is a continuous effort involving several studies [23]. Existing techniques for strengthening weak passwords broadly fall into two approaches: i) focusing on system users coming up with ways to create complex passwords which they can easily remember, and ii) password generator software is used to create more complex passwords for the system. Ultimately, the system user has to remember the construct of the password or the entire system created complex password, a burden placed on the system user by both approaches and necessitating cramming of passwords [24].

A technique in which a system user carries out a number (K) of iterations on the master password to come up with new hash has been proposed. Any adversary (e.g. hacker) would need to compute all the K hash functions for each guess in order to retrieve the original password [23]. In another technique, a password is concatenated with a random value known as a "password supplement" or "salt" before it is hashed. The strength of password is enhanced whereby any password-guessing attacker will need to perform a hash for each possible supplement until the space is searched. In other words, if the attacker has to carry out K hashes before he or she can completely eliminate that one guess as a possible password. The adverse effect of this technique lies in the fact that the regular user also needs to search through the space and needs a method for determining if the password supplement is correct [23].

4.1 Issues with Current Techniques for Strengthening User Created Passwords in EMR

User generated passwords have several weaknesses. The shorter and simpler passwords are, the more susceptible they are to being guessed and hacked by system attackers [29]. The authors further showed that 50% of user passwords are weak: derived from users' names, slang words, dictionary words for example "123456". The weakness of these passwords makes them trivial to such attacks as simple password guessing, brute force attacks and dictionary attackers. The prevalence of weak passwords has also been shown by studies indicated that up to 35% of passwords were easily cracked. In the absence of measures for enforcing password strength, most users will choose weak passwords which are even more predictable by using password recovery tools and brute force attacks [23]. The choice of weak, easy to remember passwords is often informed by the fear of being denied access to system services as a result of failure to remember complex passwords [27] [30].

It has been shown that up to 74.9% of system users have a set of predetermined passwords that they use frequently and rarely change their passwords for even more than two years if not system-forced [4]. In a related study a survey of internet users in 3 European countries (Germany, United Kingdom and Sweden) revealed that 20% of users use the same password for all online logins, 20% write down their passwords on a piece of paper where as another 8% rarely change passwords for fear of forgetting them [32].

The problem with systems that rely on password authentication is when users choose weak passwords. Weak passwords greatly contribute to compromising systems. System users have poor habits in choosing strong passwords and keeping them [13]. Weak passwords thus provide an easier penetration point leading to the compromise of more secured information systems [19]. Several tools have been developed to create strong user passwords. The main weakness with such tools is that system users are left with the burden of cramming these complex lengthy passwords. The need therefore remains for alternative mechanisms that strengthen weaker passwords with little or minimal effort required from a user's ability to remember them [17][34].

EMR systems users are therefore faced with a paradox of creating strong passwords which can't be easily guessed by attackers, and yet be able to remember them without referring to a piece of paper [27]. Users easily forget these complex and lengthy passwords especially when not used often or use the same passwords on several systems for fear of forgetting them and failing to access system services [16][31]. This has forced EMR systems users to create weak passwords they can easily remember or to use the same passwords across several sites. As a result of these weak passwords, there is a registered increase in EMR systems identity theft therefore leading to false claims that cause huge financial loss to patients and related negative impacts [27].

Users of passwords are faced with a dilemma; managing as many as 15 passwords on several sites especially with the increasing number of e-commerce sites and systems yet at the same time, they are required to recall complex and lengthy password combinations to these sites. There is a paradox between users creating complex passwords and ability to recall them without writing them somewhere yet at the same time hackers are seeking for optimizing weak passwords [15][20] [26].

For password best practices, they must be impossible to remember and never written down [23]. Too long, and complex passwords are difficult for system users to remember. Legitimate system users often face denial of service by failing to correctly login into systems due to forgetting unfamiliar lengthy passwords [16][31]. However, most humans are incapable of generating and memorizing stronger passwords. Users tend to alter characters in at times the same password for purposes of passing the sig-nup process. At times, the passwords users choose are not the best and the strongest they can afford but because they anyway have to do so in order to access a service. Often, system users forget passwords, especially more complex ones, unless written down on a piece of paper which also increases the systems security risks [27].

5 Deriving a Strengthening Technique for Weak Passwords

5.1 Determining New System Requirements

Tororo Hospital in Uganda was used for the system study. A study of the EMR system for Tororo Hospital was done. The hospital was chosen because it is way ahead in implementing the EMR systems planned to be rolled out in all the other government owned health units in the country. Existing EMR systems and processes at Tororo hospital were studied so as to gain understanding of the way users are authenticated, the strengths and weaknesses of the current practices. The focus was on user passwords, passwords management and authentication. Data from questionnaires was collected from a total of 73 respondents. These were from thirty two (32) patients, eight (8) information officers, ten (10) nurses, six (6) laboratory technicians, six (6) doctors, four (4) data officers and seven (7) office administration staff. Given the small population of possible respondents at these hospitals, a purposive sampling technique was adopted. Interviews were used to complement questionnaires and gain detailed understanding of the business processes and how user password authentication takes place. The data was analyzed in SPSS to determine requirements for the Technique to Strengthen Weak Passwords as presented in figures 2 and 3. User views on the strength of their passwords are shown in the following Fig. 2.

Fig. 2 shows responses in percentage for the responents. Overall, respondents possessed weak passwords that could not meet all the set requirements for stronger passwords. Sixty six (66%) percent of responents indicated that their passwords contain lower case characters. However, the passwords generally never met the other criterias. A majority indicated that passwords are not often changed, have no symbols, upper case characters or numbers.

User Views on EMR Security and Password Strengthening

Fig. 3 shows that a majority of system users (of EMR systems) prefer using simple, easy to remember passwords than complex ones that are not easily memorised and remembered. System users prefered the computer system to strengthen the weak password and leave them with the liberty of not memorising or cramming complex passswords.

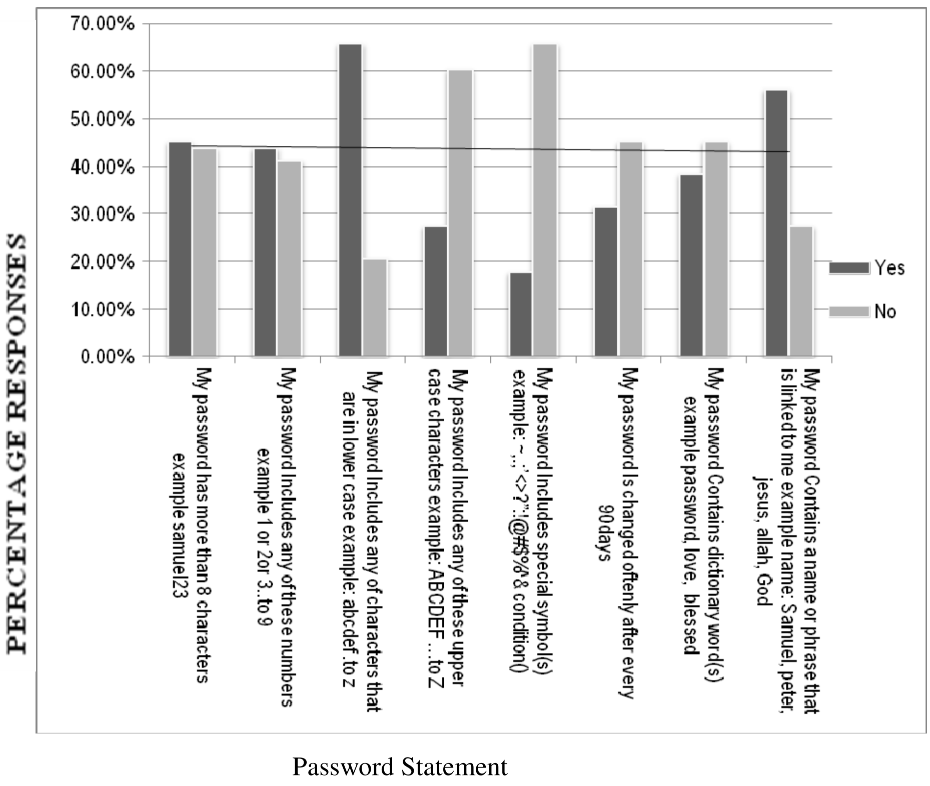


Fig. 2. Percentage Responses to Computer Password Statements

5.2 Summary of Requirements

The current system at Tororo was found to face challenges of: i) users sticking to the same password for long periods; ii) UserIDs getting publicly known and not restricted to the EMR system user and administration; iii) the login process has unrestricted password attempts; iv) encrypting passwords through use of MD5 algorithm though widely used is not secure enough. The result is the same independent of the computer used which gives a chance to guessers or hackers to carry out reverse engineering of passwords.

The study reveals the following as functional requirements for passwords: i) The UserID must not be blank and at least five (5) characters in length; ii) Users should only be granted access privileges based on their roles; iii) Login parameters should not be display on screen when one is typing them; iv) The passwords and responses to challenge questions shouldn't be stored in clear text; v) The challenge requests should be random at every login time; vi) A unique code be generated for every session expiring every time a user logs out of system; vii) Password and challenge response both should at the same time determine authentication; vii) Resulting password from strengthening process should be different at every login session.

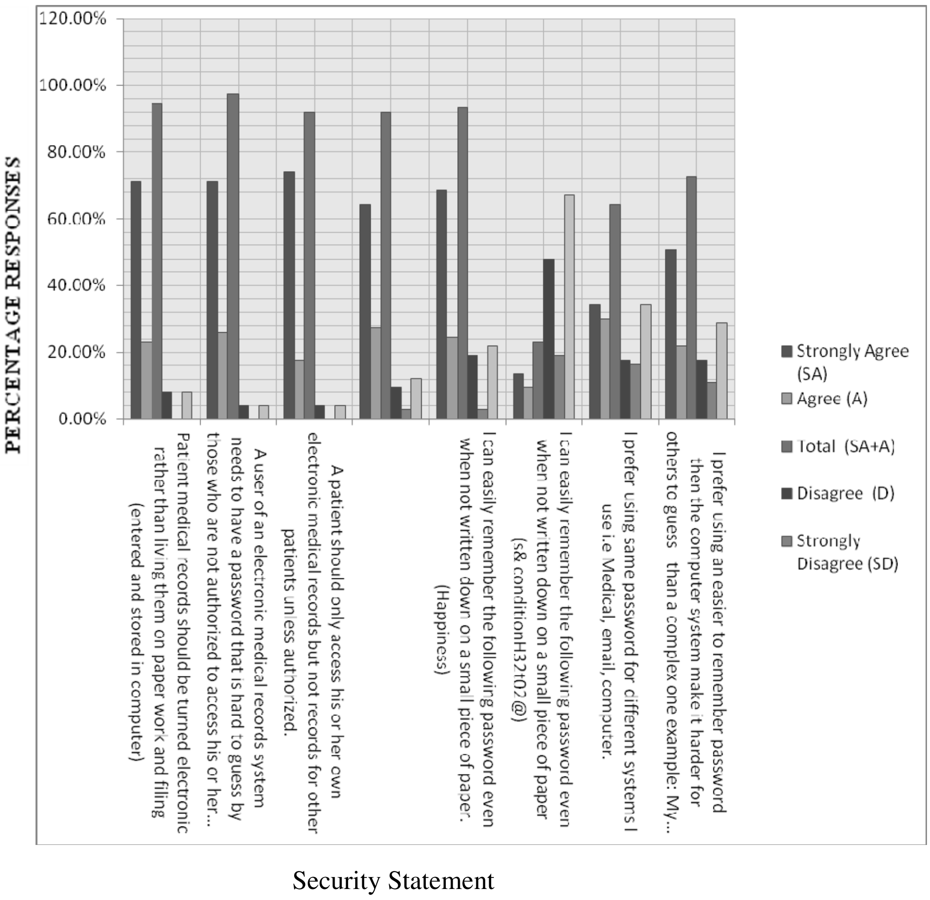


Fig. 3. Percentage responses EMR security and password strength

The study reveals the following as non functional requirements for passwords: i) The TSWP should detect that one entering the user ID is a human being not a computer; ii) The user should have entered a valid UserID then be authenticated; iii) System user enters responses to challenge questions with a minimum of five questions; iv) The users should chose their own UserID and password; v) The system user interfaces should be simple and more user friendly; vi) Length of password should be one that a system user can memorize and recall; vii) The system prototype interfaces should not be complex but simpler for user adaptation; vii) The system prototype should be able to warn a user of errors or mistakes made; vii) The system prototype should allow for the system user to open multiple windows.

5.3 High Level Architectural Design of the New User Authentication Technique

The design of a Technique to Strengthen Weak Password in EMR System (TSWP) was at the core of this study. The user Identity, Authentication and system requirements were considered when designing TSWP. Since the previous sections indicated that system users hold weak passwords the system must deploy the TSWP to come up with stronger password prior to authentication. Following authentication, the user is authorized to login.

The new design accounts for several security components at different levels of identification, authentication and authorization. When a system is started it displays a user login page which requires a user to input three parameters say UserID, Profile and dynamically system generated code. When all the three parameters meet the requirements and are valid, the system displays another login page which requires the system user to input two parameters say Password and Challenge response to a random question. When the combination of two sets parameters say UserID and Profile with Password and Challenge response results in authentic parameter (Stronger Password) same as the one in the system, a user is then granted access to the respective system components or else denied. The system user logs out and exits the system as need be.

The detailed design of the TSWP as used for EMR is given by Fig. 4. This technique is part of the UserID password authentication mechanism. Fig. 4 shows the detailed steps of user login, TSWP and the granting or denying user access to electronic records.

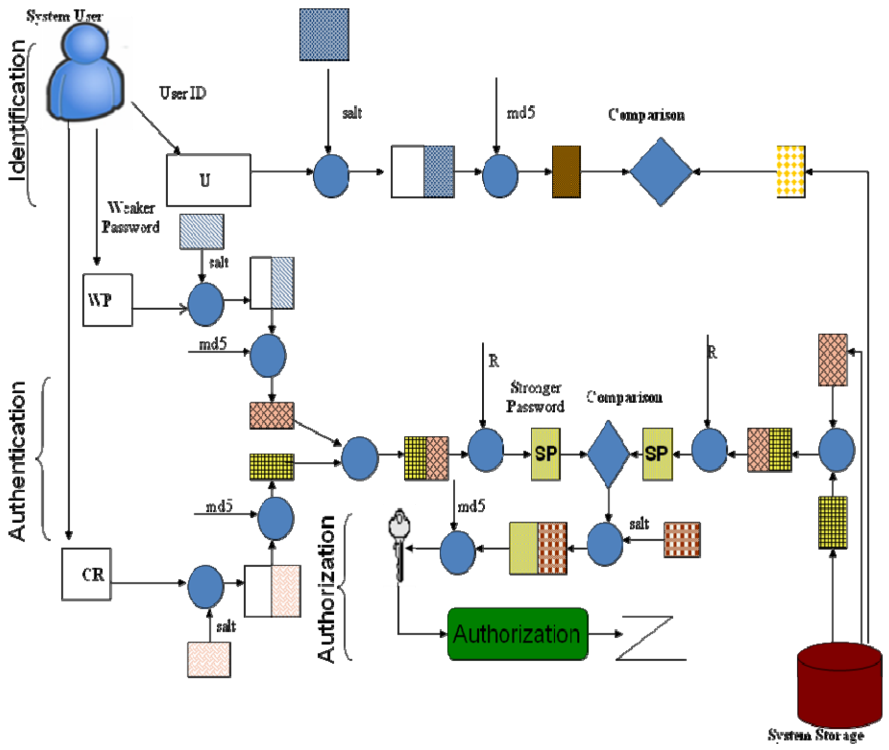


Fig. 4. Detailed Architectural Design of the Technique for Strengthening Weak Password

The TSWP is based upon three components which are password (WP), challenge response (CR), salt, random variables and MD5. When an unauthorized system user successfully provides a valid UserID, they may fail to provide the password (WP). If the unauthorized system user is successful with the UserID and password, it may not be the case with the challenge response since the requests are random. The architecture depicts three phases of Identification, Authentication and Authorization.

The salts added to WP and CR to come up with results of RST1A and RST1B before applying MD5 to yield RST2A and RST2B, make it much harder for the hacker or system attacker to perform reverse engineering of the original values. The random variables generated to replace random variables in the result (RSTC) of the concatenation of RST2A and RST2B make it much harder for an unauthorized system user to generate a stronger password (SP) since these variables change and are unique per each login. The details of each component are explained in Table 1. It shows the parameters used in the TSWP, full words, description, condition that must be met and the significance of the component.

Table 1. The information Security Components Used TSWP Architectural Design

Component/ Term	**Full**	**Description**	**Condition**	**Security Significance**
UserID	User Identification	In order for any system user to access E-Records, he or she has to possess a digital Identity here in referred to as the User Identification (UserID).	The UserID entered must be correct and valid. The UserID is case bound. UserID cant not be blank but at least six characters	User must know their UserID and if they don't, then they can't proceed to accessing E-records in the system. Case sensitivity eliminates some security threats, say eaves droppers
Profile		The category of the system user.	The profile must tarry with the UserID .	One may cram the Id but fail to know the profile therefore can't easily access E-records .
Code		Security Code that is randomly generated by the system unique for every user login.	The Code must be valid and entered correctly The Code is case bound. The Code can't be reused in another login session	This security code limits the chances of successful accessing of the system through computer bots. This provides some level surety that one logging into the system is a human being.
Password		Secret code or PIN or string of characters privately known only to an entity. The entity provides this password to prove ownership of the digital identity (UserID).	The User password must be more than 8 characters.	The password is the proof that the entity is what is claims to be or owns the digital object it claims so.
CRQ	Challenge Request	A system randomly generated question for user response.	The system user must set more than 5 questions	The randomness of the question and wider range of questions leads to more security since the unauthorized system user can't predict which question comes next.

Table 1. (*Continued*)

CR	allenge sponse	An answer that a system user enters with regard to each respective CRQ	The CRP must be valid and entered correctly The CRP is case bound.	User must know their CR and if they don't, then they can't proceed to accessing E-records in the system. Case sensitivity eliminates some security threats say eaves droppers. This adds another level of security even if the weak password is stolen or hacked. CR is used in the TSWP when coming up with a new but stronger password.
MD5	Message Digest 5	Is a one way cryptographic method that derives a digest from a message (password). The digest is a string of many characters from which it is hard to easily generate back the original message	One way cryptographic method.	MD5 is a one way cryptographic method thus not easy to derive original message other than using several messages and their digests while comparing.
Salt		String of characters added to the (Weaker) Password or Challenge response.	The salt contains a mix of characters which are hard to guess.	Since applying md5 of the same word in the same case yields the same result, it becomes much easier to do reverse engineering. The salt makes it harder for the hacker to derive the original password.
Random variables		Variables that are generated randomly from a mix of lower and upper case characters, numbers and symbols.	For every user login, these random numbers must be unique from other times. These random variables replace characters in random positions of the result (RST) of concatenation of the Weaker password and CRP.	These random variables, derived from a mix of alphanumeric characters and replacing random characters in RST, make the stronger password dynamic and stronger.
SK	Session Key		This key is unique for every user session.	Session limits access to EMR when not logged into the system. Limits the system user to their specific roles or functions. Enforces login and out of the system.
WP	Weaker Password	The password for the system user which is strengthened to come up with a stronger one (SP)	This password should be kept secret. WP must be valid and entered correctly into the system and should tarry with the UserID.	Easier for system users to remember
SP	Stronger Password	A stronger password derived from weaker password used in authenticating the system user.	This should be unique at every authentication phase.	Harder for guessing or hacking.

User authentication, in cases where the system user possesses a strong password, is based on comparing the user provided password with what is stored in the system. For this technique, in between the user's password (Weaker) and authentication lies the TSWP to strengthen this password (WP) into a stronger password (SP) before authentication takes place. A system user inputs UserID, profile and security code at identification stage. If found valid and correct the next phase is user authentication where the system user inputs the UserID and challenge response. In the next section are the steps undertaken as per the TSWP and user authentication.

Steps of the Technique for Strengthening Weak Passwords
Below are the steps that a user and a system undertake in making the weak password strong with respective examples.

a) System user inputs the UserID (example Samix), profile and security code.
Example: User inputs password (WP) samixtus and challenge response (CR) respifine.
b) When the user-entered UserID in STEP1 is valid and correct, another page is displayed which requires the user to input password (WP) and response (CR) to a randomly generated challenge request (CRQ). The two pages of user login are geared towards only authenticating valid UserIDs and the opportunity to answer derived challenge requests based on valid and correct UserIDs. The challenge questions are based on the UserID.
c) Salts are added to both WP and CR come up with new values RST1A and RST1B respectively.

These are the same salts that were used to store the WP and expected CR in the system.
Example: samixtus.98&$@3faABXT12345 and respifine.*&%@#15476abec
d) Digests of the RST1 and RST2 are done to come up with new values RST2A and RST2B. Examples:

MD5 (samixtus.98&$@3faABXT12345)	= 36806546c534e2dbdb6bc61e7df7c6fa
MD5 (respifine.*&%@#15476abec)	= 09be96f330c3a5e54bdf39606f7c499b

e) RST2A and RST2B are concatenated to come up with RST.
Example
36806546c534e2dbdb6bc61e7df7c6fa09be96f330c3a5e54bdf39606f7c499b
f) Some variables of RST are replaced with random variables to come up with a new result (RSTR). The variables in the RST are replaced randomly based on random positions. The new result is referred to as the Stronger Password (SP) which is authenticated against what the system has in order to grant user access or denial to electronic records.
Example
3#8065&6c534eAdbd16bc61e*df7c6fa09be96f330c3@5e54bbf396@6f7c799b.

5.4 Technique Validation

Validation of the TSWP system prototype was done using a group of 15 people (composed 2 patients, 4 IT staff, 6 medical records staff, 3 information security practitioners). A feedback form was supplied to the users whose results are tabulated in Table 2. The architecture of the system was also presented to the 3 information security practitioners who offered resourceful feedback that led to the final design. All 15 individuals that had time to interact with the TSWP system prototype commended the prototype and the technique behind it. Using the test and validation form, feedback was gathered from these individuals indicating that overall the requirements were met. Some of the comments from these individuals included; the technique behind the system prototype promotes security but in more creative manner. Also, it is vital to have a manual to enable users adopt the technique more easily. The system prototype also was developed with validation rules based on the requirements.

Table 2. Testing and Validation Results

R.No	**Testing and Validation Statement**	**Number of responses per category of responses**			
		SA	**A**	**D**	**SD**
R1	The system asks me to enter a security code that changes on every login.	9	6	0	0
R2	When I enter my password in a different case, the system does not allow me to access my records	8	7	1	0
R3	The system accepts my password and generates a more complex and harder to remember as well as guess password than the one I entered	11	3	0	0
R4	The system rejects invalid and incorrect passwords.	8	6	0	0
R5	The system asks me my chosen random questions and, when I correctly respond to them, then views my records.	4	9	1	0
R6	The generated password is more than eight characters	10	5	0	0
R7	The generated password contains lower case characters	11	3	0	0
R8	The generated password contains numbers	9	5	0	0
R9	The generated password contains symbols	7	6	1	0
R10	The generated password contains uppercase characters	9	6	0	0
R11	The generated password changes at every login session	10	4	0	0
R12	This new way of logging into the system is much better than entering a UserID say PitsbergDDT and password say 1B&$hg3k@3&73D09 into the system?	9	5	0	0
R13	The password generated by the system is unique at every login.	5	6	2	0
R14	The challenge questions are random at every login time.	4	9	0	0
R15	The system requires me to enter a minimum of five responses to challenge questions	6	6	1	0
R16	I recommend this new way of logging into the system to others	7	6	1	0

Some of the feedback collected was on the following questions:

a) What do you like about the way you login into the system? Answers were: i) The fact that the system first checks my name and then asks me for the password; ii) The page for inputting the UserID is looks good and tells me when I make an error; iii) I like the innovation and creativity; iv) This way of logging into the system is more interactive in that it asks me for answers to the questions I chose myself; v) If someone steals my password, it's not a guarantee that he will get to my records; vi) When I logout of the system, I can't access the inside pages without logging in again.
b) What don't you like about the way you login into the system? Answer: When I am in a rush to use the system these login levels can create impatience in me especially when I use the system many times.
c) What are some of your recommendations for improvement? Answer: The system requires training for those who are to use it, otherwise this is good innovation.

6 Discussion and Conclusions

Medical identity theft remains a challenging issue for many users of internet based electronic medical record systems. Weak password authentication mechanisms largely contribute to this problem. UserID and Password authentication has turned into a greater avenue for compromising secured information systems including (EMR system) due to weak passwords. This research therefore developed a technique for strengthening weak passwords in electronic medical record systems. Existing information security components such as salts, MD5, random variables and challenge responses were used to develop an integrated technique for strengthening weak passwords.

Challenge questions and responses are used by different companies (Yahoo, Google, Hotmail, private and public websites) as a means for one to regain a password once forgotten or for account reset purposes. Challenge questions are also used as a means of authorization of system users to particular information pertaining one's account or role when already logged into the system. The TSWP system prototype uses the result of combining the challenge response and user password when authenticating users at every logging into the system. If either of the two is incorrect and invalid, systems resources can't be accessed but at the same time the hacker finds it harder to narrow down which of the two is incorrect and invalid. Existing systems verify the challenge response independently of the password thus giving chance to a hacker to concentrate on identifying the response.

At every attempt to log onto the system, the TSWP system prototype generates random variables that are used to strengthen the product of combining the password and challenge response. Even if the salts added to the password and challenge responses are hacked, there will be a need for the hacker to get to know the particular random variables to be added to strengthen the password. The TSWP system prototype only prompts the system user to provide a password when the UserID is verified as correct and valid which is not the case with several existing mechanisms. The findings of the study identified requirements of the Technique for Strengthening Weak Passwords in Electronic Medical Record Systems which can also be used to guide similar efforts in developing similar techniques. The results of this study contribute in the following ways:

i) This research study presents insights necessary in developing Techniques for Strengthening Weak Passwords. Such techniques when developed can enable making passwords strong thereby limiting information systems security compromises resulting from weak passwords.

ii) The design of the technique can be adopted by several systems developers as a building block for securing electronic records across private and government institutions like banks, hospitals and universities but also; the same concepts can be expanded across client server network settings.

iii) The study also prototypes the way system users will not be put into the paradox of cramming and remembering complex passwords but have their weak passwords made strong.

iv) Knowledge is gained on how weak passwords can be made strong by exploring the technique devised in this study with even a system prototype.

v) As more people interact with the system and TSWP, there is continued relatively increased in trust in the security of the password authentication.

References

1. Admin. Strong passwords are especially important for government websites; Georgia Tech Procurement Assistance Center (GTPAC) (retrieved on July 28, 2010)
2. Aron, H.: Identity Theft, Password Habits, and e-Shopping Safety (2009), `http://ebay.about.com/od/ebaylifestyle/a/el_paypalstudy.htm` (retrieved on July 26, 2010)
3. Azmi, M.T., Emran, M.T.: A Survey on Computer Password Practices Undergraduate Students at Faculty of Medicine. Malaysian Journal of Community Health 12(1), 1–7 (2006)
4. Barbara, S.C.: Software Usability Research Laboratory (SURL). Wichita State University General Password Characteristics 8(1) (2006)
5. Carolyn, C.: Key Capabilities of an Electronic Health Record System (2003), `http://www.nap.edu/openbook.php?record_id=10781&page=1` (retrieved on February 2, 2009)
6. Cios, K.J., Moore, G.W.: Uniqueness of medical mining. Artif. Intell. Med. (Artificial Intelligence in Medicine) 26(1-2), 1–24 (2002)
7. Jilian, M.: Medical identity theft is on the rise and expected to worsen. Wall Street Journal (2009)
8. Dave, G., Mike, D.: A Analytics TM White Paper: Electronic Medical Records vs. Electronic Health Records: Yes, There Is a Difference Healthcare Information and Management Systems Society (HIMSS), Chicago, IL (2006)
9. Kaelber, D.C., Jha, A.K., Johnston, D., Middleton, B., Bates, D.W.: A Research Agenda for Personal Health. J. Am. Med. Inform. Assoc. 15(6), 729–736 (2008)
10. Davis, N., Chrisann, L., Kim, R.: Identity Theft and Fraud-The Impact on HIM Operations (AHIMA Practice Brief). Journal of AHIMA 76(4) (2005)
11. Dwight, O.E., Michael, R.R.: What Can Electronic Medical Records Do For You? The Journal of Lancaster General Hospital 3(4) (2008)
12. Emergis, B.: Framework for building a shared EMR (2008), `http://www.longwoods.com/product.php?productid=19603#sendtofriend` (retrieved on March 13, 2010)

13. Eugene, S.H.: Preventing Weak Password Choices. West Lafayette: Computer Science Technical Reports. Paper 875 (3) (1991), `http://docs.lib.purdue.edu/cstech/87511`
14. FDIC. Putting an End to Account-Hijacking Identity Theft (2004), `http://www.fdic.gov/consumers/consumer/idtheftstudy/identity_theft.pdf` (retrieved on April 6, 2011)
15. Halderman, J.A., Waters, B., Felten, E.W.: A convenient method for securely managing passwords. In: Proceedings of the 14th International Conference on World Wide Web, pp. 471–479 (2005)
16. James, F.L.: Password Management Strategies for Safer Systems Foil hackers. Strengthen and protect your systems' passwords. Journal of Accountancy (2009)
17. Jie, Z., Xin, L., Somasheker, A., Jennifer, Z.: Improving multiple-password recall: an empirical study. European Journal of Information Systems 18, 165–176 (2009)
18. Bell, K.M.: The National Alliance for information Technology: Report to the Office of the National Coordinator for Health Information Technology. Defining Key Health Information Technology, USA (2008)
19. Kim, Z.: Weak Password Brings 'Happiness' to Twitter Hacker (2009), `http://www.wired.com/threatlevel/2009/01/professed-twitt/` (retrieved on July 23, 2010)
20. Leslie, L., Edward, J.Y.: Password pitfalls and dynamic biometrics: Toward a multi-layer user authentication approach for electronic business. Academy of Information and Management Sciences (2004)
21. Pam, D.: The Medical Identity Theft: The Information Crime that Can Kill You. The World Privacy Forum (2006), `http://www.worldprivacyforum.org/medicalidentitytheft.html` (retrieved on July 1, 2010)
22. Lynne, R.: Cyber-Victimisation in Australia: Extent, Impact on Individuals and Responses, Curtin University of Technology, Briefing Paper no. 6 (2008)
23. Matteo, D., Pietro, M., Yves, R.E.: Password Strength: An Empirical Analysis. In: Symposium on Network Computing and Applications, Cambridge, MA, USA, July 9-11, pp. 28–35 (2009)
24. Mcafee. Techniques for strong passwords (2007), `http://www.dell.com/html/emea/` (retrieved on April 6, 2011)
25. Manoj, K.S.: Password Based A Generalise Robust Security System Design Using Neural Network. International Journal of Computer Science Issues 4(2) (2009)
26. Medlin, B.D., Crazier, J.A., Dave, D.S.: Password Selection by End Users from an eCommerce Site: An Empirical Study, p. 447 (2005), `http://aisel.aisnet.org/amcis2005/447` (retreived on July 20, 2010)
27. Mohammad, M., van Oorschot, P.C.: Digital Objects as Passwords. In: Proceedings of the 3rd Conference on Hot Topics in Security (2008)
28. Nicholas, A.K., Jonathan, B., Amit, J.N., John, G.: Electronic Medical Record Systems for Developing Countries. In: Proceedings of the International Conference of IEEE Engineering in Medicine and Biology Society (2009)
29. Redwood, S.: Imperva's Application Defense Center (ADC): Imperva Releases Detailed Analysis of 32 Million Breached Consumer Passwords (retrieved on July 27, 2010)
30. Riley, S.: Password security: What users know and what they actually do. Usability News 8(1)
31. Anderson, R.J.: Security Engineering: A Guide to Building Dependable Distributed Systems, 1st edn., p. 36. Wiley Publishing Inc. (2001)

32. San, J.: PayPal Trust and Safety Study. Identity Theft Twice as Likely in English-Speaking Countries (2008), `https://www.paypal-media.com/` (retrieved on July 16, 2010)
33. Tehan, R.: Personal Data Security Breaches: Context and Incident Summaries (Cong. Res. Serv. Rpt. RL33199) (2007)
34. Vijaya, M.S., Jamuna, K.S., Karpagavalli, S.: Password Strength Prediction Using Supervised Machine Learning Techniques. In: Proceedings of the International Conference on Advances in Computing, Control, and Telecommunication Technologies, pp. 401–405. IEEE Computer Society, Washington, DC (2009)

Author Index