David Riaño · Richard Lenz
Silvia Miksch · Mor Peleg
Manfred Reichert · Annette ten Teije (Eds.)

Knowledge Representation for Health Care

LNAI 9485

AIME 2015 International Joint Workshop, KR4HC/ProHealth 2015
Pavia, Italy, June 20, 2015
Revised Selected Papers

Springer

Lecture Notes in Artificial Intelligence 9485

Subseries of Lecture Notes in Computer Science

More information about this series at http://www.springer.com/series/1244

David Riaño · Richard Lenz
Silvia Miksch · Mor Peleg
Manfred Reichert · Annette ten Teije (Eds.)

Knowledge Representation for Health Care

AIME 2015 International Joint Workshop, KR4HC/ProHealth 2015
Pavia, Italy, June 20, 2015
Revised Selected Papers

Springer

Editors
David Riaño
Universitat Rovira i Virgili
Tarragona
Spain

Richard Lenz
University of Erlangen and Nuremberg
Erlangen
Germany

Silvia Miksch
Vienna University of Technology
Vienna
Austria

Mor Peleg
University of Haifa
Haifa
Israel

Manfred Reichert
Ulm University
Ulm
Germany

Annette ten Teije
Vrije Universiteit Amsterdam
Amsterdam
The Netherlands

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Artificial Intelligence
ISBN 978-3-319-26584-1 ISBN 978-3-319-26585-8 (eBook)
DOI 10.1007/978-3-319-26585-8

Library of Congress Control Number: 2015954352

LNCS Sublibrary: SL7 – Artificial Intelligence

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

Health-care organizations are facing the challenge of delivering high-quality services to their patients at affordable costs. These challenges become more prominent with the aging population with chronic diseases and the rise of health-care costs. The high degree of specialization of medical disciplines, huge amounts of medical knowledge and patient data to be consulted in order to provide evidence-based recommendations, and the need for personalized health care are prevalent trends in this information-intensive domain. The emerging situation necessitates computer-based support of health-care process and knowledge management as well as clinical decision-making.

For a third time, this workshop brought together researchers from two communities who have been addressing these challenges from different perspectives. The "Knowledge-Representation for Health-Care" (KR4HC) community, which is part of the larger medical informatics community, has been focusing on knowledge representation and reasoning to support knowledge management and clinical decision-making. This community has been developing efficient representations, technologies, and tools for integrating all the important elements that health-care providers work with: electronic medical records (EMRs) and health-care information systems, clinical practice guidelines, and standardized medical vocabularies. In turn, the "Process-Oriented Information Systems in Health-Care" (ProHealth) community, which is part of the larger business process management (BPM) community, focus on ways to adopt BPM technology in order to provide effective solutions for health-care process management. BPM technology has been successfully used in other sectors for establishing process-aware enterprise information systems (as opposed to collections of stand-alone systems for different departments in the organization). Adopting BPM technology in the health-care sector is starting to address some of the unique characteristics of health-care processes, including their high degree of flexibility, the integration with EMRs and shared semantics of health-care domain concepts, and the need for a tight cooperation and communication among medical care teams.

In 2012 and 2013, joint workshops were organized bringing together health-care knowledge representation as dealt with in previous KR4HC workshops, and health-care process support as addressed in previous ProHealth workshops, with a considerable success. Participants in the joint workshops could explore the potential and the limitations of the two approaches for supporting health-care knowledge and process management and clinical decision-making. The workshops also provided a forum wherein challenges, paradigms, and tools for optimized knowledge-based clinical process support could be debated. All the organizers and participants of the workshops coincided on the profit of the event, which encouraged us to organize a third edition of the joint workshop in 2015.

With the same objectives as the first and second workshop, the third joint workshop aimed to increase the interactions between researchers and practitioners from these different yet similar fields to improve the understanding of domain-specific

requirements, methods, theories, tools and techniques, as well as the gaps between IT support and health-care processes yet to be closed. This forum also provided an opportunity to explore how the approaches from the two communities could be better integrated.

Providing computer-based support in health care is a topic that has been picking up speed for more than two decades. We are witnessing a plethora of different workshops devoted to various topics involving computer applications for health care. In the last years, our goal has been to join forces with other communities in order to learn from each other, advance science, and create a stronger and larger community. The history of the two workshops, KR4HC and ProHealth, demonstrates the efforts that have been made in that direction so far.

The first KR4HC workshop, held in conjunction with the 12th Artificial Intelligence in Medicine Conference (AIME 2009), brought together members of two existing communities: the clinical guidelines and protocols community, who held a line of four workshops (European Workshop on Computerized Guidelines and Protocols (CPG2000, CPG2004); AI Techniques in Health Care: Evidence-Based Guidelines and Protocols 2006; Computer-Based Clinical Guidelines and Protocols 2008), and a related community, who held a series of three workshops devoted to the formalization, organization, and deployment of procedural knowledge in health care (CBMS 2007 Special Track on Machine Learning and Management of Health Care Procedural Knowledge 2007; From Medical Knowledge to Global Health Care 2007; Knowledge Management for Health Care Procedures 2008). Since then, five more KR4HC workshops have been held, in conjunction with the ECAI 2010, AIME 2011, BPM 2012, AIME13, and KR 2014 conferences.

The first ProHealth workshop took place in the context of the 5th International Conference on Business Process Management (BPM) in 2007. The next three ProHealth workshops as well as last year's workshop were also held in conjunction with BPM conferences (BPM 2008, BPM 2009, BPM 2011, and BPM 2014). The aim of ProHealth has been to bring together researchers from the BPM and the medical informatics communities. As the workshop was associated with the BPM conference that had never been attended by researchers from the medical informatics community, we had included medical informatics researchers as keynote speakers of the workshop, members of the Program Committee, and to our delight, saw a number of researchers from the medical informatics community actively participating in ProHealth workshops.

Following the keynote talk given by Manfred Reichert from the BPM community at the Artificial Intelligence in Medicine 2011 (AIME 2011) conference, where KR4HC was held, the organizers of ProHealth and KR4HC workshops showed interest in holding their workshops in conjunction with the BPM 2012 conference, which marked a landmark in the collaboration between the two communities. These efforts were continued when the second joint workshop took place as part of the AIME 2013 conference. Now, we are continuing these efforts with the Third Joint Workshop on Knowledge Representation for Healthcare and Process-Oriented Information Systems in Health Care (KR4HC/ProHealth).

The KR4HC/ProHealth 2015 workshop focused on IT support of high-quality health-care processes. It addressed topics including knowledge-driven health IT,

clinical guidelines and pathways, and health information systems and clinical data. Furthermore, the workshop included a special "MobiGuide project" track related to mobile process and decision-support, featuring six presentations from the FP7 MobiGuide project (www.mobiguide-project.eu).

The workshop received 24 papers from Italy (6), Israel (4), Spain (3), Canada (2), France (2), Germany (2), The Netherlands (2), Austria (1), Brazil (1), Chile (1), Poland (1), and the USA (1). Papers had to clearly establish their research contribution as well as their relation to health-care processes. Five full papers were selected to be presented at the workshop as full papers, according to their relevance, quality, and originality. One of these papers was finally retracted. The four other papers appear in this volume together with a paper by the keynote speaker. Five additional contributions submitted as full paper were selected for short presentation at the workshop, respecting the positive assessments provided by the expert reviewers. These five papers are also included in this volume.

In his keynote paper "Evolution and Revolution in Knowledge-Driven Health IT: A 50-Year Perspective and a Look Ahead," Prof. Robert Greenes from the Department of Biomedical Informatics at Arizona State University, USA, analyzed the past 50 years, in terms of what the world was like, what challenges we faced, and what achievements have been accomplished in each of the five decades since the 1960s and in our current decade. He further looked at new forces impacting us. The challenges stem from disruptions in the nature of health-care delivery, its financing, and its expanded emphasis on health and wellness as well as disease treatment, and from frequent changes in technology. One of the main disruptions for clinical decision support is that we have fragmented health systems, and communication and coordination of care are not optimal for the patient. Hospital, clinic, and home/self-care are separated and are hence not well connected. We are maximizing delivery of enterprise- or practice-specific health care rather than patient-centric care. There is limited decision-support because data are limited largely to an organization. In general, we do not have a life-time patient-oriented record with views on episodes of care and patients do not control electronic health record (EHR) usage. As another challenge, both society and the public demand more patient engagement, while at the same time facing an aging population with the world of chronic and multiple diseases growing. Hence, there is a greater need for coordination as well as for managing the continuity of care, but contemporary EHR systems do not support this. Furthermore, there are other disruptions, such as regulatory ones. In addition to these societal challenges, there is the race with technology (e.g., emerging technologies such as sensors, smart phones, and distributed apps). These are affecting the expectations of consumers, and our focus changes to meet these challenges. The original challenge of providing high-quality clinical decision support remains and has not been resolved yet. There have been success stories, but owing to the changes in technology and society expectations, the systems that have been successful are sometimes proprietary and obsolete. Moreover, newly emerging knowledge-based technologies are needed to integrate data flows and workflows across venues of care, to connect patients and the health system more effectively, and to integrate analytics to optimize decision-making. These include knowledge resources that reside outside of EHR systems and can be used to orchestrate and inform the operations of apps and services on an interoperable multi-tier platform.

The first regular paper "A Patient Simulation Model Based in Decision Tables for Emergency Shocks" by Francis Real, David Riaño, and Jose-Ramon Alonso introduced a knowledge-based simulation system hinging on decision tables about patients arriving at ICUs with shock. Seven prevalent sorts of shocks were modeled. The following four papers focus on clinical guidelines and clinical pathway support. The paper entitled "META-GLARE: A Meta-Engine for Executing CIGs" by Alessio Bottrighi, Stefania Rubrichi, and Paolo Terenziani introduces the execution component of META-GLARE, a framework to acquire, consult, and execute clinical practice guidelines represented under different CIG formalisms. The paper "Identifying Evidence Quality for Updating Evidence-based Medical Guidelines" by Zhisheng Huang, Qing Hu, Annette ten Teije, and Frank van Harmelen proposes a rule-base model to identify different levels of evidence within textual guidelines. The model can estimate the level of evidence on an average of 75 %. The paper "Answer Set Programming for Temporal Conformance Analysis of Clinical Guidelines Execution" by Matteo Spiotta, Paolo Terenziani, and Daniele Theseider Dupré describes a first approach to automate conformance checking between clinical guidelines and basic medical knowledge in order to detect contractions along time. The paper "Towards a Pathway-Based Clinical Cancer Registration in Hospital Information Systems" by Michael Heß, Monika Kaczmarek, Ulrich Frank, Lars-Erik Podleska, and Georg Täger presents an approach fostering the model-based design of process-aware health-care applications. More precisely, a domain-specific language for modeling clinical pathways is enhanced with a medical data structure (i.e., an oncologic data set) in order to enable a pathway-based (i.e., process-driven) cancer documentation in hospital information systems. The paper "A Mixed-Initiative Approach to the Conciliation of Clinical Guidelines for Comorbid Patients" by Luca Piovesan and Paolo Terenziani analyzes the technologies contained in the GLARE system to help physicians manage interactions between CIGs in order to deal with comorbid patients.

The first paper in the MobiGuide track, by Erez Shalom, Yuval Shahar, Ayelet Goldstein, Elior Ariel, Moshe Sheinberger, Nick Fung, Val Jones, and Boris van Schooten, discusses the implementation of the distributed guideline-based decision support model within the patient-guidance framework of MobiGuide. The paper presents the projection and call-back mechanism between the main backend DSS and the mobile DSS. This mechanism is used to execute projections of parts of clinical guidelines that have been customized to the requirements of concrete patients on the smart phone of the corresponding patient. The second paper in the MobiGuide track tackles the data quality problem of the mobile decision support system (mDSS) and presents the Quality-of-Data Broker, which runs on the smart phone of the MobiGuide project. The paper written by Nekane Larburu, Boris van Schooten, Erez Shalom, Nick Fung, Hermie Hermens, and Val Jones is titled "A Quality Aware Mobile Decision Support System for Patients with Chronic Illnesses." These works were presented under the context of the EU FP7 MobiGuide project.

The paper by Jens Weber, Morgan Price, and Iryna Davies addresses the problem of data quality in heath information systems. Inspired by the design-by-contract approach from software engineering, the authors propose an approach for designing and monitoring systems for various quality concerns.

We would like to thank the invited speaker as well as the members of the Program Committee and the reviewers for their efforts in selecting the papers. They helped us to compile a high-quality program for the KR4HC/ProHealth 2015 workshop. We would also like to acknowledge the splendid support of the local organization and the AIME 2015 organizers.

We hope you will find the papers of the joint KR4HC/ProHealth 2015 workshop interesting and stimulating.

September 2015

Richard Lenz
Silvia Miksch
Mor Peleg
Manfred Reichert
David Riaño
Annette ten Teije

Organization

KR4HC 2014 was organized by Richard Lenz, University of Erlangen-Nuremberg, Germany, Silvia Miksch, Vienna University of Technology, Austria, Mor Peleg, University of Haifa, Israel, Manfred Reichert, University of Ulm, Germany, David Riaño, Universitat Rovira i Virgili, Spain, and Annette ten Teije, Vrije Universiteit Amsterdam, The Netherlands.

Program Committee

Syed Sibte Raza Abidi	Dalhousie University, Canada
Ameen Abu-Hanna	AMC-UvA, The Netherlands
Luca Anselma	Università di Torino, Italy
Joseph Barjis	Delft University of Technology, The Netherlands
Arturo González Ferrer	University Carlos III of Madrid, Spain
Robert Greenes	Arizona State University, USA
Femida Gwadry-Sridhar	University of Western Ontario, Canada
David Isern	Universitat Rovira i Virgili, Spain
Stefan Jablonski	University of Bayreuth, Germany
Katharina Kaiser	University of Applied Sciences St. Poelten, Austria
Vassilis Koutkias	Institut National de la Santé et de la Recherche Médicale, France
Peter Lucas	Radboud University Nijmegen, The Netherlands
Wendy MacCaull	St. Francis Xavier University, Canada
Mar Marcos	Universitat Jaume I, Spain
Stefania Montani	University Piemonte Orientale, Italy
Bela Mutschler	University of Applied Sciences Ravensburg-Weingarten, Germany
Øystein Nytrø	Norwegian University of Science and Technology, Norway
Leon Osterweil	University of Massachusetts Amherst, USA
Hajo A. Reijers	Eindhoven University of Technology, The Netherlands
Danielle Sent	AMC/UvA, The Netherlands
Brigitte Seroussi	Hôpitaux de Paris, France
Andreas Seyfang	Vienna University of Technology, Austria
Yuval Shahar	Ben-Gurion University, Israel
Ton Spil	University of Twente, The Netherlands
Maria Taboada	University of Santiago de Compostela, Spain
Paolo Terenziani	Università del Piemonte Orientale Amedeo Avogadro, Italy
Lucinéia Heloisa Thom	Federal University of Rio Grande do Sul, Brasil
Frank van Harmelen	Vrije Universiteit Amsterdam, The Netherlands
Dongwen Wang	University of Rocherster, USA
Barbara Weber	University of Innsbruck, Austria

Contents

Knowledge-Driven Health IT and Simulation

Evolution and Revolution in Knowledge-Driven Health IT: A 50-Year Perspective and a Look Ahead

Robert A. Greenes(✉)

Arizona State University and Mayo Clinic, Scottsdale, AZ, USA
greenes@asu.edu

Abstract. In this keynote presentation, my intent is to explore the evolution of the field of computer-based clinical decision support (CDS) – and its associated tasks of knowledge acquisition, knowledge modeling, knowledge representation, knowledge management, and knowledge integration into care processes – over the past five decades. I believe that we are now in a period of significant disruption – both in the health care system itself and in the technology to support it. As a result of these disruptors, I am convinced that we will need to not only continue the efforts that have proved useful in the past but also to develop some new strategies to meet the new challenges.

1 Introduction

Many of the important ideas in CDS were discovered in the early days of the field of biomedical informatics, but a number of these could not be carried out well because technology needed to catch up. Some personal observations I will give about my experience over the years will highlight this. Our early work on hospital computer systems in the 1960s, for example, used 10 character-per-second teletype and cathode ray tube video display terminals, was implemented on computers with limited speed and 96 K (6-bit) bytes of storage, had limited (e.g., 5 MB) hard disk capacity, and relied on only telephone wire communication. An early touch screen terminal we used was homemade with aluminum strips pasted onto the screen and a capacitance circuit to detect user selections. An early hypermedia system we built in the mid-1980s for guideline navigation to aid in radiology test selection predated publicly available hypermedia authoring systems (such as Apple's HyperCard) or the WWW. Bayes theorem diagnostic programs by pioneers in the 1960s [1, 2] were built with limited databases to derive the probabilities needed.

Given limitations such as these, it was often difficult to determine whether slow progress in various application realms, particularly in CDS, was due to technology limitations or flaws in the basic concept or approach. But, as discussed in several chapters in [3], the slow progress in CDS adoption over the past 50 years, with primary successes being in computer-based provider order entry (CPOE)/order sets, infobuttons, alerts and reminders, and, to some extent, documentation templates, and to a lesser extent, execution of clinical guidelines, makes us further reflect on this question. Even today, those primary successes are actually quite modest, in terms of the degree to

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 3–20, 2015.
DOI: 10.1007/978-3-319-26585-8_1

which these approaches to providing CDS are actually used – mainly in academic or large enterprise-owned health systems with necessary staff to tailor and customize them, and barely at all in small practice and community hospital or clinic settings. This makes us wonder whether the basic paradigms are the right ones for scaling up to widespread adoption and use, or whether we should reconsider the whole concept of what CDS means and what is needed in the evolving and future health system.

We will briefly trace the decades of evolution of computing technology and methodology for integration of knowledge. I will provide some examples from personal experience. We will then consider our current period of disruption, and project how the evolution will continue and how and where more revolutionary change may be called for.

2 Conflicting Paradigms

The dominant paradigm of the past 50 years was one in which electronic health record (EHR) systems gradually evolved and became the primary vehicle through which computer applications in health care were provided. These began as hospital-based systems in the 1960s, gradually extending to ambulatory care facilities, and only much later to smaller office practices. In the 1990s and beyond, regional consolidation of hospital networks and affiliated practices occurred in the United States and some other countries. In general, this was specific to particular enterprises, and there were often competing practices and enterprises in many regions. In all cases, the focus was on the information and communication needs of the provider or hospital office, facility, or enterprise, and on local optimization of that entity, in terms of workflow, efficiency, and financial returns. Reimbursement of providers and hospitals was largely on a fee-for-service basis, so optimization often was seen in terms of maximizing volume and minimizing costs per transaction.

In the late 1990s early 2000s, based in large part on two landmark studies by the Institute of Medicine [4, 5], recognition of the frequency of medical errors as a major cause of patient deaths and of uneven and inadequate quality of care began to shift the emphasis to safety and quality as well as efficiency. Reimbursement strategies seeking to reward safety and quality were difficult to implement, largely because of the inability to track outcomes over time and to determine appropriate levels of care for particular conditions. The shift in more recent years to recognition of the importance of wellness, prevention of disease, and early and aggressive management of disease, to avoid or delay complications, has led to a change in the focus from primarily on improving a *health care system* to establishing a *health system* – with many more players (patient, caregiver, community, as well as traditional health care entities) – and to increased emphasis on patient-centered, rather than practice-centered care.

This means that our EHR systems are poorly suited to the new paradigm, because care often crosses enterprise boundaries, and EHR systems also do not necessarily maintain lifetime continuous records of patients (although in some settings they do of course achieve this). Personal health record (PHR) systems have not evolved robustly either, for lack of a strong business model for them. Further, an emphasis on wellness and prevention can lead to reducing the need for acute or high-intensity care and hospital

use, and is associated with incentives for home monitoring, email and phone communication of providers with patients, and other activities that fee-for-service for care processes traditionally have not paid for. In the health system paradigm, the goal is to reward providers and patients for doing less and achieving better results, so tying financial rewards to volume and intensity of services is exactly backwards.

Thus we have today a health care system that is different from where we began, but with EHR systems built for a model that is now 50 years old and even inappropriate. The EHR systems are often themselves 20–30 years old, with proprietary architectures, data structures, and knowledge content, not readily amenable to integration and continuity of care across transitions of care.

3 A Selective Five-Decade Historical Review from a Personal Perspective

This review is not intended to be comprehensive but will reflect trends that occurred from the days when I first became active in the field of biomedical informatics, which coincided with the birth of the field itself. So I am including some reflections on my own work as illustrations of the kinds of activities that were occurring during these periods.

3.1 The 1960s

The earlies decision support was aimed at diagnosis. Ledley and Lusted's classic paper [6] in 1959 introduced Bayes' theorem for medical diagnosis and led to several interesting projects by early investigators such as Warner et al. and Lodwick, cited above. Interestingly, Ledley and Lusted first introduced in this paper also decision theory/decision analysis as a method of selecting pathways with the maximum expected value (later called utility), although interest in this did not pick up until perhaps 15–20 years later.

In the 1960s another form of decision support was algorithmic calculation, e.g., for drug dose estimation and for acid-base/electrolyte balance management [7]. Specialized programs implementing these were built, but there was also some early work on creating shells or drivers for applications. For example, in my own work, I built a series of tools in the 1960s for (a) statistical data processing [unpublished systems in use at Massachusetts General Hospital (MGH) in Boston], (b) Geographic Information Systems (GIS)-based mapping [8], (c) neurophysiologic signal analysis [unpublished internally sponsored project at MGH], and (d) guideline-based consultation, teaching, or interviewing [unpublished, submitted as honors thesis for MD degree at Harvard]. These had the model in which a method was provided and the data and knowledge driving a particular use were external and incorporated into the programs at execution time, to determine its particular function.

As work on hospital information systems kicked off, we also were faced with the early technology limitations mentioned earlier. Minicomputers were just becoming available and able to run in a time-sharing mode with some of the earliest time-sharing software. But there weren't any good high-level languages for minicomputers, and writing hospital

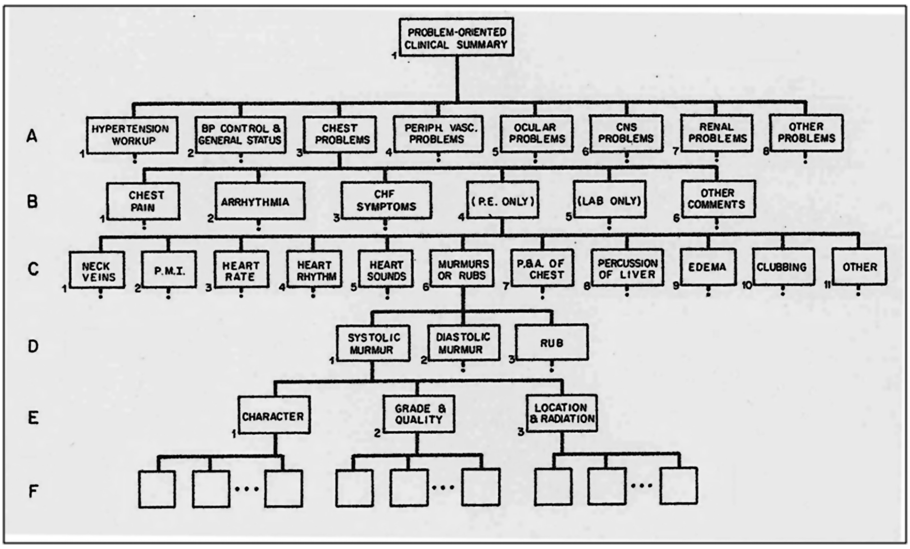


Fig. 1. The content schema used for knowledge-guided entry of progress notes for hypertension care, 1969. See [10].

information system applications in assembly language, compiling, checking acceptability to users, and responding to errors and feedback took weeks instead of minutes. To address these issues, we actually needed to build whole new operating systems, not just shells and drivers. The MUMPS system developed by us at Massachusetts General Hospital mid-1960s [9] was an effort to make it easy to quickly generate interactive applications, run them interpretively, and provide a flexible persistent data store for patient data. Another shell with an embedded knowledge structure was also built by us in the 1960s using MUMPS, for managing the knowledge underlying hypertension (see Fig. 1), to drive an interactive progress note entry system in the hypertension clinic of MGH [10]. This was carried out as a test of the problem-oriented record formalism, as the idea of problem-oriented records was being introduced by Weed [11]. [It also served as a basis for my PhD thesis at Harvard.] The latter used the homemade touch screens I mentioned earlier (see Fig. 2). We had no inklings of graphic display terminals and GUIs at that time, and the interfaces were clunky, slow, and not very flexible, yet we were able to demonstrate reasonable success in creating a feasible interaction mode for reporting. It was also used in experiments in radiology report capture [12].

3.2 The 1970s

In the 1970s, there was some gradual advance in hospital systems. Alerts and reminders were introduced by McDonald [13]. This was also the decade of AI – early rule-based systems pioneered by Shortliffe [14], and other heuristic diagnostic systems such as Internist-1 [15], and the Present Illness Program [16].

Fig. 2. The homemade touch-screen interface to an interactive display terminal used for progress note entry in 1969 hypertension progress note structured data capture project. See [10].

One of the important ideas that got further developed in the 1970s was the separation of the knowledge base from the inference engine, e.g., E-Mycin [17], although also done for example in the earlier consultation drivers/shells we built in the 1960s, and other applications that used external knowledge and data to drive them. This gave rise also to a series of studies about knowledge acquisition, knowledge representation, and knowledge management.

We began using this approach to integrating CDS into clinical systems, with the conceptual model shown in Fig. 3. The separation may only be virtual, in the sense that the CDS may be within the same computing environment, but the separation enhances modularity and reusability, and is thus a valuable design approach in any case. Later service-oriented architecture models (in the 2000s) make the separation more formal, e.g., [18].

3.3 The 1980s

In the late 1970s and early 1980s, more attention was placed on guidelines and decision models. Decision analysis came into its heyday [19]. EHRs expanded in their scope of

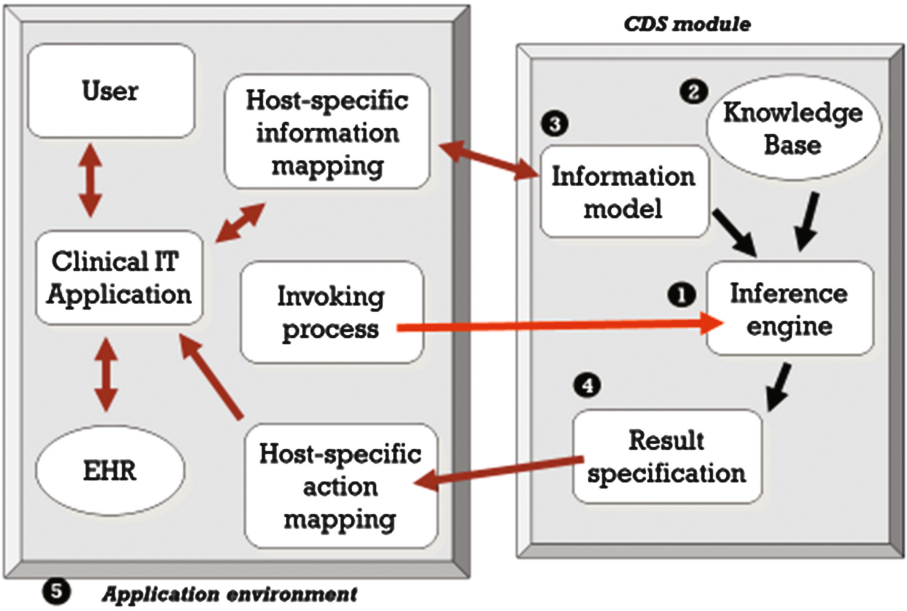


Fig. 3. A conceptual model of the separation of CDS inference engines and knowledge bases from the invoking processes, used as a basis for our ongoing CDS projects.

ancillary subsystems encompassed, and network communication enabled them to begin to exchange data. We built some guideline navigation tools. Microcomputers were just becoming available and networks of workstations were being explored. Graphic user interfaces were available. We built a system to support obstetrical ultrasound examinations by digitizing images, measuring fetal structures, computing fetal age and development based on a model using those measurements, and enabling entry of other findings to generate a report [20]. We built an early hypermedia system for radiology procedure appropriateness guidelines [21] (see Fig. 4). We also began to develop a driver for multimedia integration at the desktop [22] and some Bayes theorem visualization tools [23].

This was also the period in which the Unified Medical Language System (UMLS) work began [24], and we served as a contractor for that work, building some early browsing and navigation tools for semantic classification of concepts [25, 26].

3.4 The 1990s

In the 1990 interest in AI turned to data mining and predictive modeling, given the difficulty in scalability of rule-based and heuristic knowledge bases. We also saw growth of the Internet, and the introduction of the WWW. We did some early work on systems for integrating distributed components at the desktop, using CORBA technology [27] and gradually moved this work to web services as the latter became available. We created some of the early systems for use of the web as a platform for

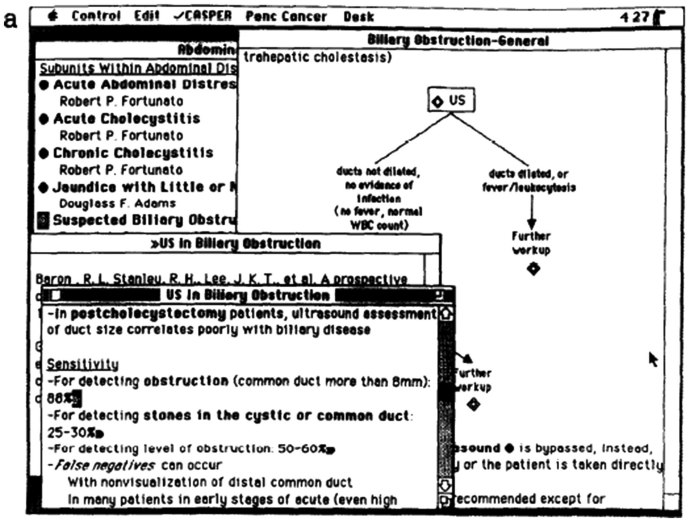


Fig. 4. Examples of multimedia guideline content presented on an early Macintosh computer in the 1980s.

hospital based intranet and internet content management [28], for journal publishing [29], and for health knowledge services to patients [30].

At the same time, there was growing interest in alerts and reminders and standards for rules (Arden Syntax) [31] and for terminologies and messages. We began to work on guideline authoring with several collaborators [32, 33]. GLIF was born of those efforts [34–36].

3.5 The 2000s

In the 2000s the focus shifted to enterprise health care systems, efficiencies, safety and quality, and the work on guidelines increased as did attempts to standardize them. We also saw concerns growing about the knowledge explosion, especially as genomics and precision medicine became prominent.

Thus more standards work (e.g., attempts to standardize guideline languages, which ultimately, and to this day, were not successful, and the establishment of a standard for the expression language in GLIF [37], and work on distributed platforms, architectures, health information exchange, and data warehouses expanded. And work on knowledge management systems began to grow. A study we did at Partners Healthcare focusing on a knowledge inventory and a follow-up study on the multiple rule subsystems with incompatible renditions of rules focused on the need for a knowledge management system and possible external services such as rules engines [38]. This led to the launch of a Partners knowledge management initiative, with similar efforts done in some other major healthcare enterprises.

During the whole period to this point, we had been seeing gradual increase in knowledge-based applications and services. CDS was most prominent in the areas of order entry, order sets, alerts and reminders, and infobuttons (see several chapters in [3]). Some guideline use and some documentation templates have also occurred. Most success was in academic and large enterprise systems, with very little penetration until recently elsewhere. This was due in part to the local implementations, lack of pervasiveness of standards, proprietary tools, and lack of incentives to share. It was also the result of difficulty in tailoring to workflows and in optimizing use for busy providers in ways that enhanced rather than reduced efficiencies.

4 The Current Period: Mid-2000s Onward – A Time of Multiple Disruptions

In the mid-to-late 2000s and continuing into the present decade, the disruptors shown in Fig. 5 have all begun to arise. Some of these have their roots earlier, but each of those has now itself become a significant driver for change; together they make for a "perfect storm" of forces that will drive expanded use of CDS, but also require different ways of delivering and managing the knowledge underlying it.

We will not discuss these disruptive forces in detail, but it is sufficient to say that they have multiple origins – social, cultural, policy-driven, financial, and technical – and there is some overlap among them. It is interesting to ask how much technology is the driver or a response to changing requirements. It is certainly the case that both are true. It is hard to imagine a world without smartphone technology now, but many of the

	Science	Technol.	Policy	Stds.	Soc. trends
1. Precision medicine	X				
2. Biosensors		X			
3. Patient engagement					X
4. Big data	X	X	X		
5. Pay for value			X		X
6. Wellness & prevention			X		X
7. Regulatory demands			X	X	
8. Usability needs			X		X
9. Rise of an app culture		X			X
10. Interoperability		X	X	X	
11. Augmented guidance		X		X	X

Fig. 5. Eleven factors that are serving as disruptive forces of our current health care system, and of the IT system that serves it. The columns indicate spheres in which these disruptive forces are exerting their influence. The brackets on the left indicate that there is some overlap among these disruptors. (Adapted from [39] with permission).

functionalities now embedded in smartphones were not only unachievable previously but were not even thought of. Innovation is partly driven by what is possible, not just what is needed. Ideas are unleashed by new technical possibilities.

Also, we might ask whether gradual change is really a disruption. If something becomes smaller and more powerful and less expensive, year after year, this is a gradual change, but at some point, the size and form factor and portability and affordability become so significantly different that a quantitative change has led to a qualitative change in dimensions such as: where, for what purposes and by whom the item can be used. Similarly if health care costs, the aging of the population, or the amount of knowledge needed to provide care keep expanding, qualitative changes in the nature of the care process, priorities, or resource availability may arise. I believe we are in this kind of situation with many of the changes in Fig. 5.

This is also illustrated by the differences in the topics covered in the first and second editions of the book *Clinical Decision Support,* which I have had the privilege of editing, the 2007 version (*The Road Ahead*) [40] and the 2014 version (*The Road to Broad Adoption*) [3]. Because of the impact of genomics, big data, personal and wearable devices, patient engagement, a shift to focus on wellness, apps opening up the possibilities for improved visualization and usability, evolution of standards and interoperability, and other factors, the opportunities for and kinds of decision support have greatly expanded. Between the two editions, in a mere 7 years, the book's increased in size by 40 % with eight new chapters and two chapters no longer relevant.

The main point about all of these changes is that they are shifting our priorities to a health system with emphasis on:

- continuity of care
- communication among health care participants
- smooth transitions of care
- optimal workflows among the participants in the conduct of care
- connecting the patient – including wearables and sensors

so that the patient is the center of the process over his/her lifetime. Further, given the accumulation of data from many sources, there is increased effort to normalize and to identify the data flows and pipelines for creating archives of patient-specific data (see Fig. 6), and from those, aggregate databases for research, quality measurement, population measurement, and other purposes. The integration of predictive analytics into the care process is emerging as a major area of activity as we evolve to what is now often referred to as a "learning health system" [41].

Thus we are now in a period where the old paradigms of proprietary EHRs need to give way to models and platforms of a patient-centric record – whether virtual or real – and where EHRs and other data sources are contributors to this record and able to view and use aspects for enterprise-specific tasks and work processes, but where the control of the data will lie with the patient. This transition will not be easy, but gathering forces in the form of standards initiatives, regulations, health care organization-led consortia, and other efforts are beginning to exert increasing pressure on vendors to enable interoperability at multiple levels. This has several implications which we and others have been exploring, as we consider the appropriate platform for health and health care going forward (see also [39]). It is clear that it will need to be a multi-tiered platform

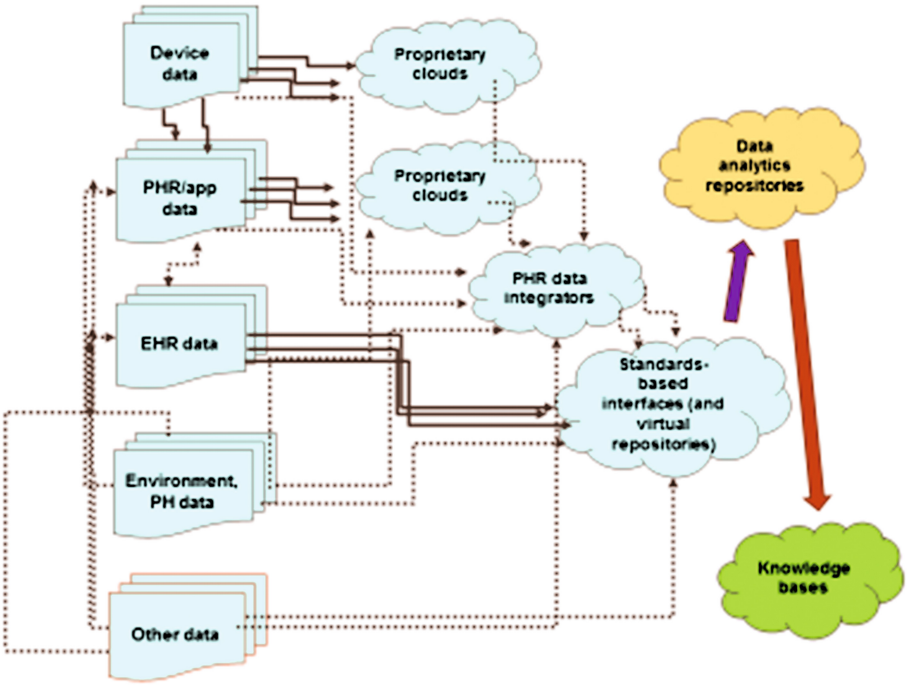


Fig. 6. Depiction of some of the multiple sources in which clinical and health-related data arise, and the data flows/pipelines which need to be aligned if we are to be able to achieve a lifetime, continuous view of a patient's health/healthcare status. This depicts also the aggregation of data for analytics, and the derivation of knowledge from the data. (Adapted from [39] with permission).

with data sources such as EHRs and patient records and device data repositories, and data warehouses all connected. Standard interfaces such as HL7's FHIR [42] are beginning to be adopted to enable consistent access to data. Intermediate-tier services for master patient index, record locator, security, authentication and role-based authorization, context management, and other functions are needed. In addition a cascade of knowledge-based services are needed – external to the EHRs, which is a change from the current model where much of this is embedded in the EHRs – to provide terminology, normalization, classification, and other higher-order semantic relations, and to support business processes, workflows, and orchestration among apps and services. Apps and suites of apps can be viewed as sitting at an upper layer atop this set of services and resources.

A project from Harvard known as SMART introduced the idea of an "app store" for healthcare. They built SMART apps which interface with EHRs for visualization and display of patient data in interesting ways [43, 44]. However, there is need for more than this kind of single, self-contained, read-only apps. The growing need to execute care pathways and protocols that extend across venues of care and sometimes across enterprise boundaries, e.g., from home to provider office to hospital to specialist or

imaging center or to extended care, and to ensure that data and problem lists and care plans are up-to-date and available to all care participants, requires a set of external knowledge services.

Other demands include the ability to integrate data from patients, wearables and sensors, and to maintain a lifetime patient-centered record, whether virtual or real, as the primary source of patient care, with EHRs transitioning to creating views into and enterprise-specific annotations and business processes concerning these data rather than being the primary sources of the data.

Aggregate data across patients and big data analytic techniques will increasingly inform the care process, from population management techniques for identifying high risk, high utilizer, or inappropriate care subgroups that need particular attention, to predictive modeling, to direct retrieval of patients similar to a current patient in order to provide analytic support to a decision about the patient's treatment options.

All of these functions require new levels of ability to integrate a diverse range of data, knowledge and services, and develop a set of apps to present and organize this information for users – patients, providers, other care givers, administrators, public health personnel, etc.

As we seek to enable this broad range of functionality in the future, it is hard to conceive of these capabilities occurring simply by expanding existing EHRs and adding connectivity and information exchange at the edges. Rather, we need to enable a broad platform of data sources, intermediate services and resources, and an upper tier of well-designed apps suited to particular needs.

Further, we don't refer to a middle tier of a three-tiered architecture but rather to a multi-tiered platform with several layers of knowledge-based componentry. We envision workflow processes and orchestration of services and apps being knowledge-driven. Apps themselves can be knowledge-driven, and the connections and communications among them can be knowledge-driven.

Thus we see a need for a much expanded knowledge infrastructure with a range of knowledge tools – for semantic relation management, inference, rule processing, workflow management, and other functionality, and authoring and knowledge management tools to support them. Moreover, these will need to be built and shared using standard modeling approaches, and able to be shared and documentable and transparent.

5 Enabling a New "Ecosystem" for Health IT

How does such a new health IT platform come about? I believe that we will eventually get to such a system architecture, given the growing desire for integrated, connected health, data liquidity, and functional interoperability. But the time course can be greatly influenced by several factors. Our current IT system has had a 50-year history, as we have discussed. I hope we do not have to wait another 50 years for the desired future platform to emerge. The following may be some important drivers for getting there faster:

1. Compelling use cases – ideally driven by major, well-recognized entities in health care or by government or other large-scale initiatives. The use cases we have articulated – continuity across transitions of care, connected care engaging with

patients and their devices, and integrating analytics to inform the care process – are all tasks not done well by existing EHR-based approaches. Key organizations we are working with in my own group include Mayo Clinic, Intermountain Healthcare, Louisiana State University Medical Center, and the Veterans Administration, all seeking to address these use cases.

A major push in the United States for Accountable Care Organizations (ACOs) [45] is also an important driver. ACOs are groups of providers including primary care, specialists, imaging centers, hospitals, and other entities all working together to manage patients, where their reimbursement is tied to patient outcomes rather than to fee-for-service. In other words, they are paid to keep people well, to use less-intensive services, and to keep patients out of the hospital. EHRs don't typically include all the necessary data for optimal care and the workflow processes across transitions of care are not well integrated, and there are insufficient data for determining desired outcomes as a basis for payment or for measurement of performance of providers against outcome for particular diseases. Thus this is a very important use case that is growing rapidly in the U.S.

2. Consortium power. Although individual entities, particularly large ones, may have some influence on EHRs and other products, greater leverage can be achieved by forming consortia of the health care providers and other key stakeholders, including vendors but taking care in setting up the consortium not to be dominated by proprietary interests. Such consortia can push for and help develop standards and reference architectures and platforms, test beds, and showcases of solutions for important use cases. The Healthcare Services Platform Consortium (HSPC) with which my group is working is one such example, involving a number of the aforementioned participants [46].
3. Sandbox/testbed environments. It is very difficult for a developer to get started in creating a solution for this new ecosystem without assembling the necessary infrastructure, tools, services, test data, and other resources. My group, working with participants in the HSPC, is establishing a sandbox environment that provides a multi-tiered platform reference implementation and enables students, faculty, and other collaborators, including small and large companies, to develop and test either (a) modifications to the infrastructure such as new services, (b) knowledge resources, or (c) apps that rely on the infrastructure. Any of these can become refined to a point that they can be offered to the broader community, either as open source contributions or as commercial offerings. Coupled with the sandbox we expect to offer business development/"incubator" services. (See Fig. 7).

Part of the environment we created is an app development and deployment framework known as AppWorks, which enables apps to be constructed from individual widgets that perform specific functions, and which are composed into app suites through knowledge-based event triggers and filters, and a context manager [47]. The widget library is extensible, and we seek to have it grow through contributions by developers. (See Fig. 8).

Fig. 7. A multi-tiered standards-based architecture being developed for supporting apps and suites of apps operating on top of multiple data sources including EHRs. This indicates the intermediate tier services that are assembled to provide the platform. Classes of services are shown with candidate particular services indicated. (Adapted from [39] with permission, and modified courtesy of D. Sottara, Arizona State University).

6 Knowledge Resource Needs

A key aspect of the new platforms is the externalization of the knowledge resources controlling the orchestration of apps and services, managing business processes and workflows, and providing decision support. Considering the history we reviewed, this will be a significant departure from the past. Although trends to separating knowledge from inference engines and the development of external decision support and query answering (e.g., infobutton) services have continued, most of the management of data, its translation, its post-coordination into useful clinical information models, and the invocation of higher level inferencing processes, assembly of documentation templates and order sets, and processing of rules have been done within EHR systems. As a result, much of the tooling, editing and authoring environments, and representation models have been proprietary or embedded in the EHRs.

Fig. 8. Examples from the AppWorks design and deployment framework that we have been developing for use on a multi-tiered platform such as that shown in Fig. 7. Apps are composed of widgets ("applets") in page layouts or sequences of pages. Widgets communicate with each other with knowledge-based event invocations, and can impose filters on the kinds of knowledge communicated. Widgets are all context-aware through context sharing features of the architecture.

What we need for future challenges such as described above for an integrated health IT ecosystem is to move to is an environment where knowledge resources live in

common libraries and there is shared tooling for managing data flows, inferencing, orchestrating workflows, managing context, and enabling CDS.

Work such as the Health e-Decisions (HeD) initiative sponsored by the U.S. Office of the National Coordinator for Health IT (ONC) [48] has resulted in a formal model for decision support artifacts, based on upper ontologies and specialized sub-ontologies, and a rich set of metatags. My group contributed to this model and built an editor for it [49] which has the virtue that, because of the formal modeling foundation, artifacts could be converted into various delivery forms for serialization or for execution. The idea of a shared repository of executable knowledge was promoted by various initiatives such as the Clinical Decision Support Consortium, a project funded by the U.S. Agency for Healthcare Quality and Research for about 8 years until the mid-2010 s [50].

I participated in organizing a consortium known as the Morningside Initiative [51] at around the same time, with some funding from the U.S. Department of Defense, to provide shared knowledge representations and tools. With other funding through ONC, that project later shifted to be part of the HeD modeling and authoring initiative. Our current work is on extending this model, and the U.S. Veterans Administration has issued contracts to colleagues of ours to further develop it for its use. As part of the HSPC, we are now looking at extending this model as a framework for managing knowledge artifacts needed for the knowledge-driven multi-tiered platform discussed above.

An interesting aspect of the future of knowledge-guided care can be seen as an extrapolation of the above, in terms of what I refer to as “augmented guidance”. It is analogous to the Global Positioning Systems (GPS) units we use for navigating our physical environment. Consider that a future health IT system is able to maintain continual context awareness and situation awareness of participants (providers, patients), so that any encounter can be classified by the patient problem, provider role and expertise, the setting, current trends, goals, and care plans. This should enable continual instantaneous retrieval of relevant knowledge resources, based on the metatags and ontology classifications of the resources matching a current context and situation. Function would be very similar to the GPS ability to retrieve relevant local attractions and services or suggest alternate routes based on expected hazards in the direction one is heading. This would reduce the need to explicitly build triggers for all forms of decision support in all individual settings, since much of it could be anticipated by context and situation tags.

7 Conclusions

Our journey in Health IT has been a long one, with evolution occurring over the past 50 or more years. Many innovations have developed over the years, some not scalable or practical until technology was able to catch up. The evolution of systems has reached a point, however, where it is in some ways mismatched with the evolution of the overall health care system and its transition to a more holistic perspective of a health system. Many disruptors are at play that suggest that, beyond evolution of the IT system, we need more fundamental change in the health IT platform.

We have described some driving use cases of major health care entities with whom we are working, and some efforts to organize and move the largely entrenched ecosystem in this direction. It is not clear how long such an evolution will take, but it is clearly inevitable. Various strategies and policies may hasten or slow down this evolution. Our own efforts are focused on building models, platforms, sandbox environments, and pilot projects based on the driving use cases. As part of the platform development, it is clear that external knowledge resources, beyond those embedded in proprietary EHR systems, will be needed. This has become an important focus of our current work.

References

1. Warner, H.R., Toronto, A.F., Veasy, L.G.: Experience With Baye's theorem for computer diagnosis of congenital heart disease. Ann. N. Y. Acad. Sci. **115**, 558–567 (1964)
2. Lodwick, G.S.: A probabilistic approach to the diagnosis of bone tumors. Radiol. Clin. North Am. **3**, 487–497 (1965)
3. Greenes, R.A.: Clinical Decision Support: The Road to Broad Adoption, 2nd edn. Elsevier, New York (2014)
4. Kohn, L., Corrigan, J., Donaldson, M.: Committee on Quality of Health Care in America IoM. To Err Is Human: Building a Safer Health System. National Academies Press, Washington, D.C. (1999)
5. IOM. Crossing the quality chasm: a new health system for the 21st century. National Academy Press, Washington D.C. (2001)
6. Ledley, R.S., Lusted, L.B.: Reasoning foundations of medical diagnosis; symbolic logic, probability, and value theory aid our understanding of how physicians reason. Science **130**, 9–21 (1959)
7. Bleich, H.L.: Computer evaluation of acid-base disorders. J. Clin. Invest. **48**, 1689–1696 (1969)
8. Greenes, R.A., Sidel, V.W.: The use of computer mapping in health research. Health Serv. Res. **2**, 243–258 (1967)
9. Greenes, R.A., Pappalardo, A.N., Marble, C.W., Barnett, G.O.: Design and implementation of a clinical data management system. Comput. Biomed. Res. **2**, 469–485 (1969)
10. Greenes, R.A., Barnett, G.O., Klein, S.W., Robbins, A., Prior, R.E.: Recording, retrieval and review of medical data by physician-computer interaction. N. Engl. J. Med. **282**, 307–315 (1970)
11. Weed, L.L.: Medical records that guide and teach. N. Engl. J. Med. **278**, 593–600 (1968)
12. Pendergrass, H.P., Greenes, R.A., Barnett, G.O., Poitras, J.W., Pappalardo, A.N., Marble, C. W.: An on-line computer facility for systematized input of radiology reports. Radiology **92**, 709–713 (1969)
13. McDonald, C.J.: Protocol-based computer reminders, the quality of care and the non-perfectability of man. N. Engl. J. Med. **295**, 1351–1355 (1976)
14. Shortliffe, E.H., Davis, R., Axline, S.G., Buchanan, B.G., Green, C.C., Cohen, S.N.: Computer-based consultations in clinical therapeutics: explanation and rule acquisition capabilities of the MYCIN system. Comput. Biomed. Res. **8**, 303–320 (1975)
15. Miller, R.A., Pople Jr., H.E., Myers, J.D.: Internist-1, an experimental computer-based diagnostic consultant for general internal medicine. N. Engl. J. Med. **307**, 468–476 (1982)
16. Pauker, S.G., Gorry, G.A., Kassirer, J.P., Schwartz, W.B.: Towards the simulation of clinical cognition. Taking a present illness by computer. Am. J. Med. **60**, 981–996 (1976)

17. van Melle, W., Shortliffe, E.H., Buchanan, B.G.: EMYCIN: a knowledge engineer's tool for constructing rule-based expert systems. In: Buchanan, B.G., Shortliffe, E.H. (eds.) Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project, pp. 302–313. Addison Wesley, Reading (1984)
18. Kawamoto, K., Honey, A., Rubin, K.: The HL7-OMG Healthcare Services Specification Project: motivation, methodology, and deliverables for enabling a semantically interoperable service-oriented architecture for healthcare. J. Am. Med. Inform. Assoc. **16**, 874–881 (2009)
19. Schwartz, W.B.: Decision analysis. N. Engl. J. Med. **301**, 54 (1979)
20. Greenes, R.A.: OBUS: a microcomputer system for measurement, calculation, reporting, and retrieval of obstetric ultrasound examinations. Radiology **144**, 879–883 (1982)
21. Greenes, R.A.: Computer-aided diagnostic strategy selection. Radiol. Clin. North Am. **24**, 105–120 (1986)
22. Greenes, R.A.: "Desktop knowledge": a new focus for medical education and decision support. Methods Inf. Med. **28**, 332–339 (1989)
23. Greenes, R.A.: Interactive microcomputer-based graphical tools for physician decision support. Aids to test selection and interpretation and use of Bayes' theorem. Med. Decis. Making **3**, 15–21 (1983)
24. Lindberg, D.A., Humphreys, B.L., McCray, A.T.: The unified medical language system. Methods Inf. Med. **32**, 281–291 (1993)
25. Sato, L., McClure, R.C., Rouse, R.L., Schatz, C.A., Greenes, R.A.: Enhancing the Metathesaurus with clinically relevant concepts: anatomic representations. In: Proceedings of the Annual Symposium on Computer Applications in Medical Care, pp. 388–391 (1992)
26. Greenes, R.A., McClure, R.C., Pattison-Gordon, E., Sato, L.: The findings–diagnosis continuum: implications for image descriptions and clinical databases. In: Proceedings of the Annual Symposium on Computer Applications in Medical Care, pp. 383–387 (1992)
27. Greenes, R.A., Deibel, S.R.: Application development in the DeSyGNER environment: dynamic assembly of independent information resources via direct manipulation authoring. In: Proceedings of the Annual Symposium on Computer Applications in Medical Care, pp. 913–915 (1991)
28. Karson, T.H., Perkins, C., Dixon, C., Ehresman, J.P., Mammone, G.L., Sato, L., Schaffer, J.L., Greenes, R.A.: The PartnerWeb Project: a component-based approach to enterprise-wide information integration and dissemination. In: Proceedings of the AMIA Annual Fall Symposium, pp. 359–363 (1997)
29. Shareck, E.P., Greenes, R.A.: Publishing biomedical journals on the World-Wide Web using an open architecture model. In: Proceedings of the AMIA Annual Fall Symposium, pp. 343–347 (1996)
30. Greenes, R.: Harvard's HealthAware project guides patients through healthcare decisions. Internet Healthc. Strateg. **3**, 4–5 (2001)
31. Hripcsak, G.: Arden syntax for medical logic modules. MD Comput. **8**(76), 78 (1991)
32. Peleg, M., Boxwala, A.A., Tu, S., Zeng, Q., Ogunyemi, O., Wang, D., Patel, V.L., Greenes, R.A., Shortliffe, E.H.: The InterMed approach to sharable computer-interpretable guidelines: a review. J. Am. Med. Inform. Assoc. **11**, 1–10 (2004)
33. Shortliffe, E.H., Barnett, G.O., Cimino, J.J., Greenes, R.A., Huff, S.M., Patel, V.L.: Collaborative medical informatics research using the Internet and the World Wide Web. In: Proceedings of the AMIA Annual Fall Symposium, pp. 125–129 (1996)
34. Ohno-Machado, L., Gennari, J.H., Murphy, S.N., Jain, N.L., Tu, S.W., Oliver, D.E., Pattison-Gordon, E., Greenes, R.A., Shortliffe, E.H., Barnett, G.O.: The guideline interchange format: a model for representing guidelines. J. Am. Med. Inform. Assoc. **5**, 357–372 (1998)

35. Peleg, M., Boxwala, A.A., Ogunyemi, O., Zeng, Q., Tu, S., Lacson, R., Bernstam, E., Ash, N., Mork, P., Ohno-Machado, L., Shortliffe, E.H., Greenes, R.A.: GLIF3: the evolution of a guideline representation format. In: Proceedings of the AMIA Symposium, pp. 645–649 (2000)
36. Boxwala, A.A., Peleg, M., Tu, S., Ogunyemi, O., Zeng, Q.T., Wang, D., Patel, V.L., Greenes, R.A., Shortliffe, E.H.: GLIF3: a representation format for sharable computer-interpretable clinical practice guidelines. J. Biomed. Inform. **37**, 147–161 (2004)
37. Sordo, M., Boxwala, A.A., Ogunyemi, O., Greenes, R.A.: Description and status update on GELLO: a proposed standardized object-oriented expression language for clinical decision support. Medinfo **11**, 164–168 (2004)
38. Greenes, R.A., Sordo, M., Zaccagnini, D., Meyer, M., Kuperman, G.J.: Design of a standards-based external rules engine for decision support in a variety of application contexts: report of a feasibility study at Partners HealthCare System. Medinfo **11**, 611–615 (2004)
39. Greenes, R.A.: Health Information Systems 2025. In: Weaver, C., Ball, M., Kim, G., et al. (eds.) Healthcare Information Management Systems: Reality, Visions and Future Roadmaps, 4th edn. Springer, Switzerland (2016)
40. Greenes, R.A.: Clinical Decision Support: The Road Ahead. Elsevier, New York (2007)
41. Friedman, C.P., Wong, A.K., Blumenthal, D.: Achieving a nationwide learning health system. Sci. Transl. Med. 2: 57cm29 (2010)
42. FHIR. FHIR DSTU 1 (V0.0.82): Welcome to FHIR (2014). http://www.hl7.org/implement/standards/fhir/. Accessed 21 March 2015
43. Mandl, K.D., Mandel, J.C., Murphy, S.N., Bernstam, E.V., Ramoni, R.L., Kreda, D.A., McCoy, J.M., Adida, B., Kohane, I.S.: The SMART Platform: early experience enabling substitutable applications for electronic health records. J. Am. Med. Inform. Assoc. **19**, 597–603 (2012)
44. SMART. SMART: Tech Stack for Health Apps (2015). http://docs.smarthealthit.org/. Accessed 21 March 2015
45. Lowell, K.H., Bertko, J.: The Accountable Care Organization (ACO) model: building blocks for success. J. Ambul. Care Manage. **33**, 81–88 (2010)
46. HSPC. The Healthcare Services Platform Consortium (2015). https://healthservices.atlassian.net/wiki/display/HSPC/Healthcare+Services+Platform+Consortium. Accessed 29 July 2015
47. Greenes, R.A., Boxwala, A.: AppWorks - An innovative design and delivery tool for interoperable health apps (2015). https://sites.google.com/site/appworkshealth/home. Accessed 21 March 2015
48. HeD. Health eDecisions Homepage (2014). http://wiki.siframework.org/Health+eDecisions+Homepage. Accessed 21 March 2015
49. Greenes, R.A., Sottara, D., Haug, P.J.: Authoring and Editing of Decision Support Knowledge. In: Zhang, J., Walji, M. (eds.) Better EHR: usability, workflow and cognitive support in electronic health records. University of Texas, Houston, Houston (2014)
50. Middleton, B.: The clinical decision support consortium. Stud. Health Technol. Inform. **150**, 26–30 (2009)
51. Greenes, R., Bloomrosen, M., Brown-Connolly, N.E., Curtis, C., Detmer, D.E., Enberg, R., Fridsma, D., Fry, E., Goldstein, M.K., Haug, P., Hulse, N., Hongsermeier, T., Maviglia, S., Robbins, C.W., Shah, H.: The Morningside Initiative: collaborative development of a knowledge repository to accelerate adoption of clinical decision support. Open Med. Inform. J. **4**, 278–290 (2010)

A Patient Simulation Model Based on Decision Tables for Emergency Shocks

Francis Real[1], David Riaño[1(✉)], and José Ramón Alonso[2]

[1] Research Group on Artificial Intelligence, Universitat Rovira i Virgili, Tarragona, Spain
{francis.real,david.riano}@urv.net
[2] Emergency Department, Hospital Clínic de Barcelona, Barcelona, Spain

Abstract. Physicians in Intensive Care Units (ICU) have to deal with shocked patients very often. These are critical emergencies that require rapid and precise clinical reactions in order to avoid fatal organ decline and patient's death. New treatments cannot be checked directly on real patients because of the risks this could imply. For this reason several simulators have been published. These simulators use to be implemented as complex mathematical models and they are oriented to concrete types of shocks or focused on fluid resuscitation. Here, we present a new simulator which is based on decision tables. It is able to simulate the evolution of seven different sorts of shocks when treated with fluid resuscitation and vasoactive agents.

Keywords: Simulation · Shock · Decision tables · Knowledge representation

1 Introduction

Shock is a common condition in critical care, affecting about one third of patients in the intensive care units (ICU). It is described as the clinical expression of circulatory failure that results in inadequate cellular oxygen utilization [1].

Some of the most common shocks are cardiogenic shock, anaphylactoid shock, cardiac tamponade, hemorrhagic shock, neurogenic shock, shock due to acute pulmonary embolism, and septic shock.

Clinical reaction to shocks in ICU must be fast and precise because of the vital consequences on the patient and to prevent worsening organ dysfunction and failure. These reactions entail the combined application of ventilatory support, fluid resuscitation, and vasoactive agents [1].

All these actions have a direct and sometimes immediate consequence in some internal hemodynamic parameters. These parameters combine under the name of cardiac output, and they are: volemia (or the amount of fluids), heart rate, contractility (or heart strength), and vasoconstriction (or the weight of the vessels).

Since many of these parameters are not directly observable by the physician who is attending the patient, medical decisions must be taken in terms of some

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 21–33, 2015.
DOI: 10.1007/978-3-319-26585-8_2

observable vital signs such as: heart rate, central venous pressure, arterial blood pressure, systolic blood pressure, diastolic blood pressure, finding of hematocrit, and superior vena cava oxygen saturation.

The capacity to foresee the consequences of medical interventions in patients with shock can reduce the risks associated to this medical problem. In this sense, several simulators have been implemented for shocks. So, Arturson et al. [2] describe a mathematical model based in 19 differential equations, 147 algebraic equations, and about 150 variables. Roa et al. [3,4] propose a non-linear macroscopic mathematical model for patient fluid distribution during the first 48 h after injury is presented. Dagan et al. [5] incorporate the Sheffer's model [6] and baroreflex [7] in a multi-layer system, providing a mathematical model for hemodynamic, oxygen balance, and control mechanism.

All these simulators are oriented to burned or bleeding patients, that are two cases of the hypovolemic shock and they are focused on fluid resuscitation exclusively. This implies two major limitations: on the one hand, the existing models are single-shock oriented, therefore several independent models are needed if we want to deal with multiple shocks, but integrated models are preferred in ICU's where a response to all sort of shocks must be given. On the other hand, as far as we are aware, existing models are centered in fluid resuscitation for shock, which represents only one part of the real treatment provided at ICU's.

In order to overcome these two limitations, we constructed a simulator able to represent not only hypovolemic shock, but also distributive, cardiogenic and obstructive shocks in an ICU. Our model will consider both fluid resuscitation and vasoactive agents and it will represent the knowledge about shock hemodynamics as a variation of decision tables [8,9].

Our simulator relies on initial emergency treatment of patients with shock arriving to an ICU, accordingly to the signs and clinical actions mentioned in clinical practice guidelines [1,10–18].

Decision tables are knowledge structures in which columns represent rules, and rows represent either conditions (antecedents of the rules) or actions (consequents of the rules). They have been qualified as intuitive, simple, fast, flexible, clear, and powerful structures [19,20]. These features make decision tables very suitable to represent knowledge coming from medical experts and its subsequent validation [21,22].

The rest of the paper is organized as it follows: in Sect. 2 we formalize the basic information and knowledge structures that define a treatment of the seven sorts of shock considered. In Sect. 3, we describe the simulator in terms of the structural design. The knowledge contained inside the simulator is explained in Sect. 4. Then in Sect. 5 we discuss the model and provide the conclusions of this work.

2 Formalization of the Treatment of Shock

The treatment of shock is based on three main aspects: ventilate (i.e., oxygen administration), infuse (i.e., fluid resuscitation), and pump (i.e., administration of vasoactive agents).

These are three sorts of treatment actions that, combined together, aim to control a set of non-observable hemodynamic parameters (cardiac output) that manifest in terms of some observable vital signs. Formally speaking, we have a set of signs and symptoms (S) that describe the patient condition with regard to shock, and a set of clinical actions (A) which can be continuous or discrete depending on the duration of their effects on the hemodynamic parameters. Continuous actions have effect while they are applied, but the effect disappears when the action is interrupted. On the contrary, discrete actions have persisting effects over time. From a medical point of view, in an emergency context (few hours), we can assume that the effect of discrete actions persists along the whole patient simulation time at the ICU.

During the treatment, the patient may evolve along a sequence of states (P_i), being P_0 the condition of the patient when admitted, and P_m the condition of the patient at discharge time. The evolution of a patient is then seen as a sequence $<P_0, P_1, ..., P_m>$, where each P_i defines the values observed for the signs and symptoms in S at the i-th stage of the patient evolution. Each P_i is a subset $\{(S_j^i, U_j^i)\}_j$, where S_j^i is a sign in S, and U_j^i the value of S_j^i for the patient in state P_i.

Simultaneously, the treatment can be adjusted as the patient evolves. The complete treatment T on a concrete patient is then a temporal sequence $<T_1, T_2, ..., T_m>$, where each T_i defines the clinical actions performed when the patient was in condition P_{i-1}. Each T_i is a subset $\{(A_j^i,V_j^i)\}_j$ where A_j^i is one of the actions in A, and V_j^i can be "take", "do not take", or a dosage, with regard to the clinical action A_j^i. Moreover, a time δ_i exists between the application of T_i and T_{i+1}. For example, when a patient arrives with an anaphylactoid shock, with Systolic and Diastolic blood pressure 72/39 mmHg P_0 is $\{SBP = 72, DBP = 39\}$), then the physician may decide to give epinephrine 1mg IV bolus (i.e., $T_1 = \{(EB,1\,mg)\}$), which causes the patient to evolve to a new state with a blood pressure 95/58 mmHg ($P_1 = \{SBP = 95, DBP = 58\}$), after $\delta_1 = 10$ m.

In order to construct a patient simulator for cardiogenic shock, anaphylactoid shock, cardiac tamponade, hemorrhagic shock, neurogenic shock, shock due to acute pulmonary embolism, and septic shock, we have used the clinical guidelines [1,10–18] to identify a set of signs and symptoms S containing seven vital signs: heart rate, central venous pressure, arterial blood pressure, systolic blood pressure (SBP), diastolic blood pressure (DBP), finding of hematocrit, and superior vena cava oxygen saturation. In addition, our simulator will be sensitive to a set of clinical actions A containing 17 different actions: antihistamine, hydrocortisone, epinephrine bolus, atropine, diuretic, fluid infusion, plasma transfusion, red blood cell packed, dopamine infusion, dobutamine infusion, norepinephrine infusion, epinephrine infusion, vasodilators, thrombolytic therapy, reperfusion (KT), pericardiocentesis, and insertion of intra-aortic balloon counterpulsation[1].

[1] Clinical actions of the sort infusion are considered continuous. The rest are discrete.

3 The Patient Simulation Model

The evolution of patients affected of some of our seven targeted shocks when they are subject to fluid resuscitation and vasoactive agents is reproduced with a Patient Simulation Model that uses the formalization introduced in the previous section. This Patient Simulation Model (PSM) has two different modules. The first one is the Action Module (AM) which deals with calculating the changes that would be produced in a theoretical standard patient if a set of treatment actions were applied. AM is where simulation holds. The second module, called the Patient Module (PM), is able to define patient cases with different features in order to interact with the simulator. Both modules are described in the next subsections.

3.1 The Action Module

When a patient receives several treatment actions, her hemodynamic parameters (cardiac output) may be affected, and this can provoke some of her vital signs to suffer a modification. The primary task of AM is to calculate the effect of one or more treatment actions in the patient vital signs. Every single action may affect one or more vital signs, and the combination of several actions may affect the same vital sign in many different ways.

In order to calculate how the treatment actions modify the vital signs of the patient, the AM module uses a variation of decision tables [21]. Regular decision tables are knowledge structures representing rules as columns. The premises of the rules appear as rows at the top of the table, while the conclusions appear as rows at the bottom of the table. Conditions in the premises appear in the table cells intersecting the column of the rule and the row of the corresponding premise. Conclusions in the rule are marked with an X in the corresponding conclusion row of the column representing the rule.

We proposed a modification of the structure of regular decision tables so that the new structure could contain the main simulation rules required to implement AM. See Table 1. These new tables have as inputs the current vital signs of the patient (rows $vitalSign_i$, for all $vitalSign_i$ in S), and the treatment actions performed (rows $treatmentAction_i$, for all $treatmentAction_i$ in A). The output of these tables are the increment or decrement expected for each vital sign (rows $vitalSign_i$ at the bottom).

Table 1 shows a generic example of such sort of decision tables. One of such tables exists for each shock in the simulator.

In the table, the first inputs represent vital signs. We can represent Yes/No, enumerated, and numeric signs. Yes/no signs such as $VitalSign_1$ can contain values of the sort Yes or No. Enumerated signs such as $VitalSign_2$ can contain labels representing numeric intervals (e.g., very high, high, medium, low, or very low) and numeric signs such as $vitalSign_s$ can contain explicit numeric intervals (e.g., $>= 90$, < 120, or $90 - 120$). Tables can also contain unknown values ($-$) as in $Rule_n$.

Table 1. Generic example of decision table in the Action Module

	$Rule_1$	$Rule_2$		$Rule_n$
$VitalSign_1$	Yes	No	...	–
$VitalSign_2$	*medium*	*low*	...	*high*
...				
$VitalSign_s$	$>= 90$	< 120	...	$90 - 120$
$TreatmentAction_1$	Yes	No	...	Yes
$TreatmentAction_2$	high dosage	low dosage		medium dosage
...				
$TreatmentAction_t$	No	No	...	Yes
$VitalSign_1$	$+15$	$+5$	...	
$VitalSign_2$	-6	-10	...	$+10$
...				
$VitalSign_s$		$+0.4$	...	-0.1

The last inputs in the decision table are the treatment actions. They can contain Yes/No values (e.g. $TreatmentAction_1$), used to indicate whether the treatment follows a clinical procedure or not, but also enumerated values (e.g. $TreatmentAction_2$), used to indicate dosages as high dosage, low dosage, or medium dosage. Other dosage granularities are also possible.

The output of decision tables represents the modifications of the vital signs caused by treatment actions or by combinations of treatment actions. The modification can represent an increment (+), a decrement (−) or a null effect (empty cell) for each vital sign. Increments and decrements are associated a relative magnitude of the change (e.g., +5 or −0.1), +15 meaning that it has triple incidence in the vital sign than +5.

During the simulation process, one or more rules of the table can be activated simultaneously. In this case, all the values are added to the corresponding vital sign. For example, if two rules activate with respective incidences +15 and −5 on a vital sign S_j^i with a current value U_j^i, the new value for that sign after δ_{i+1} time will be U_j^{i+1}=$U_j^i + 15 - 5$.

3.2 The Patient Module

Under the same health condition, different patients can have different normality parameters for their vital signs, and their response to a same treatment can vary. For example, a SBP = 100 could be considered normal for a certain patient, but very low for a patient with hypertension because the normality parameter of these two patients for SBP are different. Also, some patients may present resistance to certain drugs or hypersensitivity to some treatments. Sometimes, the general health condition of a patient or her risk factors (which are not necessarily

related to the shock under consideration) can make certain clinical actions not to be recommended or even counter-indicated. Additionally, the same dosages may have different effects depending on each clinical condition.

These are some of the reasons why the application of AM alone is incomplete to implement a correct simulator. So, we complemented our simulator with a Patient Module (PM) that adapts the results provided by AM to the patient under consideration.

While the AM can be seen as running the simulation for a standard patient and calculating a standard response, the PM works to simulate a customization of the results in accordance to the features of each single patient.

In the PM we are allowed to determine the special behavior of the patient for each treatment action in A by defining her sensitivity/resistance with a percentage: 0 % representing full resistance, values between 0 % and 100 % partial resistances, 100 % the standard effect, and values above 100 % crescent sensitivities. See an example in Table 2 with patient sensitivities in the section `ACTION SENSITIVITIES`.

Furthermore, comorbidities and physical conditions may vary vital signs normal references and their limits, in every single patient. Clinicians use this sort of variations to determine the treatment goals. In the PM we are allowed to specify normality parameters of cases by means of ranges that will be used not only to assess the effects of clinical actions over tolerable limits but also to decide on ICU discharges. See these ranges for a case example under `VITAL SIGN RANGES` in Table 2.

In PM, cases are allowed to contain sensitivity/resistance percentages for all the 17 clinical actions related to the shocks, and vital sign ranges for all the 7 vital signs relevant to the shocks under consideration. Table 2 describes a case of a 72-year female admitted with a septic shock with risk factor hypertension and other secondary diseases. Her normality parameters are defined under the section `VITAL SIGN RANGES`, with boundaries `MIN`, `LOW`, `HIGH` and `MAX`. These values define the ranges for unacceptably low (below `MIN`) and unacceptably high (above `MAX`) causing the simulation to stop, at risk (between `MIN` and `LOW` or between `HIGH` and `MAX`) requiring urgent intervention at the UCI, and normal (between `LOW` and `HIGH`) to consider ICU discharge. The patient also shows 130 % hypersensitivity to resuscitation using intravenous fluid and 80 % sensitivity (i.e., 20 % resistance) to norepinephrine effects. The section `INITIAL SIGNS` describe the values for all the sings of the case at the time of admission in the ICU.

3.3 The Iteration System

The combined action of AM and PM provides simulation of a case under some particular circumstances. This simulation is punctual in time, but shock management uses to be a process consisting of several steps. In order to provide "long term"[2] simulations representing patient evolution, we propose an iteration system.

[2] The concept long term must be understood in the context of an ICU (few hours).

Table 2. Example of patient definition with vital sign ranges and clinical action sensitivities

```
Patient ID : 12
AGE : 72    SEX : female    WEIGHT : 65.0 Kg   HEIGHT : 1.57 m
SHOCK : Septic shock

DESCRIPTION:
Medical history: Arterial hypertension. No previous medications.
Diagnostics: biliary septic shock due to cholecystitis
             acute renal failure

VITAL SIGN RANGES:          MIN     LOW   HIGH    MAX    UNITS
Systolic blood pressure    40.0    85.0  140.0   280.0    mmHg
Diastolic blood pressure   20.0    50.0   90.0   130.0    mmHg
Arterial blood pressure    26.6    61.6  106.7   180.0    mmHg
Heart rate                 25.0    60.0  100.0   148.0    bpm
Central venous pressure     0.0     3.0    8.0    20.0    cmH2O
Finding of hematocrit      15.0    35.0   45.0    60.0      %
Sup vena cava oxygen sat   50.0    65.0   85.0    88.0      %

ACTION SENSITIVITIES:
Resuscitation using intravenous fluid  ==>  130 %
Norepinephrine                         ==>   80 %

INITIAL SIGNS:
  <Systolic blood pressure = 62.0>
  <Diastolic blood pressure = 38.0>
  <Arterial blood pressure = 46.0>
  <Heart rate = 118.0>
  <Central venous pressure = 0.1>
  <Finding of hematocrit = 47.0>
  <Sup vena cava oxygen sat = 64.0>
```

The external appearance of the iteration process starts with the initial set of vital signs values for the patient (i.e., P_0 or the values of `INITIAL SIGNS` in Table 2) and the definition of the sensitivity/resistance percentages for that case (section `ACTION SENSITIVITIES`). Then, a set of clinical actions (T_1) is applied, this causing some modifications in the state of the patient (P_1) after a time δ_1 and according to both the rules in the AM decision table of the shock under consideration and the sensitivity/resistance values of the case in PM. This is repeated till a discharge state (P_m) is reached.

Delta times (δ_i) simulate the times that physicians have to wait in order to the changes in the patient states become effective. These times may vary between 15–20 m in emergency treatments and weeks in other clinical treatments.

These delta times are narrowly related to the response times associated to the clinical actions in A. The response time of an action is defined as the time for the

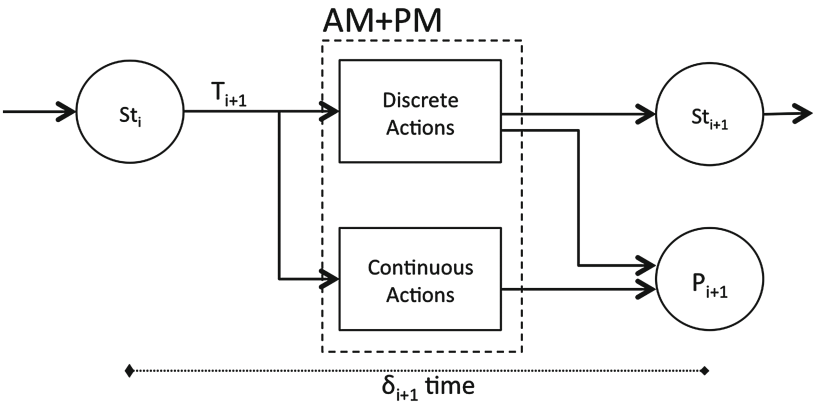


Fig. 1. Iterations with AM and PM

effects of the action to be observed. All the clinical actions in our simulator have a response time that ranges between immediate response (e.g. pericardiocentesis) and 30 m (e.g. thrombolytic therapy).

This external appearance of the iteration process has an internal implementation in the simulator that Fig. 1 depicts.

Here, the state of the patient at admission (P_0=St_0) is treated with the set of clinical actions T_1. This set may contain both, continuous actions and discrete actions. Continuous actions have an effect in the immediate next state of the patient. On the contrary, discrete actions have an effect that persists for the whole ICU treatment, as for example the administration of some drugs like epinephrine bolus or the application of a medical procedure like the insertion of intra-aortic balloon counterpulsation. Some procedures can only be applied once along the treatment.

In our simulation of shocks, continuous clinical actions are dopamine infusion, dobutamine infusion, norepinephrine infusion, and epinephrine infusion. The rest of actions (antihistamine, hydrocortisone, epinephrine bolus, atropine, diuretic, fluid infusion, plasma transfusion, red blood cell packed, vasodilators, thrombolytic therapy, reperfusion (KT), pericardiocentesis, and insertion of intra-aortic balloon counterpulsation) are discrete, being the last four procedures.

The application of both continuous and discrete actions in T_{i+1} to the patient transforms (some of) her vital signs obtaining a new patient state (P_{i+1}) which is observable by the users of the simulator. This process is the result of applying AM and PM to the pair (St_i, T_{i+1}). However, another internal state (St_{i+1}) of the patient is calculated by the simulator. This new state describes the vital signs of the patient from a global perspective required by the simulator to continue with a new iteration of the simulation process, after a time δ_{i+1}. This internal state is the result of applying AM and PM to the pair (St_i, D_{i+1}), with D_k=$\{A_j^k \in T_k$: A_j^k is a discrete action$\}$.

In order to calculate δ_i we take the largest response time of the clinical actions in T_i.

4 Application of the Patient Simulation Model

The above model has been applied to the construction of a simulator for patients affected of one of the following shocks: cardiogenic shock, anaphylactoid shock, cardiac tamponade, hemorrhagic shock, neurogenic shock, shock due to acute pulmonary embolism, and septic shock.

The construction of the AM decision tables was carried out in cooperation with senior physicians of the Emergency Department of the Hospital Clínic de Barcelona. The implications of each action in the vital signs were studied for each one of the shocks. The medical experts were consulted through examples about the expected effects that each one of the actions should have (e.g. *If a patient has anaphylactoid shock, a heart rate of 90 bpm, and she takes antihistamine, then what the new expected value for the heart rate is?*). More than 1000 examples of clinical conditions were analyzed for each shock. The results were analyzed to define the rules that were included in the decision tables. For each shock a decision table was constructed with an average of 50 rules (i.e., columns) per table.

Table 3. Extract of Decision table for Septic Shock

	rule 1	rule 2	rule 3	rule 4	rule 5	rule 6
SBP	< 85			< 85		
DBP		< 50			< 50	
HR			> 100			$60 - 100$
Fluid infusion	Yes	Yes	Yes			
Norepinephrine infusion				$0.12 - 0.6$	$0.12 - 0.6$	$0.12 - 0.6$
SBP	$+5$			$+25$		
DBP		$+1$			$+10$	
HR			-10			$+15$

Table 3 shows an extract of the decision table for the septic shock. We use this table to illustrate the next case example that correspond to the patient described in Table 2: A 72-year, 65 Kg female arrives to the ICU with a biliary septic shock (i.e. gallbladder infection) and initial vital signs: systolic blood pressure (SBP) 62 mmHg, diastolic blood pressure (DBP) 38 mmHg, and heart rate (HR) 118 bpm, among others (see `INITIAL SIGNS` in Table 2).

The physician decides to deliver fluids (500 ml saline solution). The simulator (AM) activates rules 1 to 3 and calculates that, after 20 m, the new vital signs will be: SBP 67 mmHg, DBP 39 mmHg, and HR 108 bpm. These 20 m are calculated as the response time of the fluid infusion applied. Since the case shows default sensitivity (100 %) to fluid infusion (note that no special sensitivity/resistance is indicated for fluid infusion in Table 2), the calculated vital sign values are not modified by the PM.

With this new information on the state of the patient, the physician concludes that more fluids are required, and continues with other saline solution. After a new delay of 20 m, the new vital signs are calculated to be: SBP 72 mmHg, DBP 40 mmHg, and HR 98 bpm. This is the result of applying the same rules 1 to 3 in the decision table.

In front of this new state, the physician decides to give norepinephrine infusion at 0.2 mcg/Kg/min. Now rules 4, 5 and 6 in the Table 3 conclude that after 15 m (response time for norepinephrine infusion), the patient's vital signs should evolve to SBP 97 mmHg, DBP 50 mmHg, HR 113 bpm. But, since the patient is 80 % sensitive to norepinephrine (see Table 2), the simulated evolution of the patient with AM+PM will be SBP 92 mmHg, DBP 48 mmHg, HR 110 bpm. Notice that these values are the result of calculating 80 % of the values +25, +10, and +15 (i.e., +20, +8, and +12, respectively) and modify previous DBP, SBP, and HR values with these increments.

Since none of the vital signs is at a normal level for the patient (see `VITAL SIGN RANGES` in Table 2), the shock intervention at the ICU should continue.

5 Discussion and Conclusions

Simulation is a technique to "replace or amplify real experiences with guided experiences that evoke or replicate aspects of the real world" [23]. The term "simulator" used in health care usually refers to a device that presents a simulated patient and interacts appropriately with the actions taken by the simulation participant [24]. Barjis et al. [25] have classified health care simulation in four areas: Clinical Simulation, Operational Simulation, Managerial Simulation, and Educational Simulation.

Clinical Simulation is used to study, analyze and replicate the behavior of diseases and biological processes in human body. *Operational Simulation* is used for capturing, analyzing and studying health care operations, service delivery, scheduling, health care business processes, and patient flow. *Managerial Simulation* is used as a tool for managerial purposes, decision making, policy implementation, and strategic planning. *Educational Simulation* is used for training and educational purposes.

Operational Simulation and Managerial Simulation are closely interrelated and correspond to the components for health care process management. Conversely, Clinical Simulation and Educational Simulation are more related with the patient care, and the sort of simulator that we have described in this paper.

Shock treatment has two important features that make it a special case in ICUs. These are, the need of a rapid intervention and the vital risks of the clinical decisions. Simulators of shocked patients are tools that can allow physicians to have more calm, reflexive, and risk-free performances [26]. All such simulators that we are aware of have complex internal mathematical models whose adaptation to new evidences and to the changes in clinical guidelines is complicated and it puts some difficulties to their evolution as new computer versions. It is also worth to mention that the knowledge behind these simulators use to be

hidden behind complex mathematical formalisms or distribution privacy rules, that make the physicians using these simulators unable to fully understand the reasoning of the simulator and also prevent them from participating in the modification or extension of that knowledge.

In addition, these sort of simulators are single-shock oriented and therefore difficult to use in a combined way to manage patients with more than one shock, for example cardiogenic shock whose treatment causes an anaphylactoid shock.

In spite of the complexity of the shock simulators that we are aware of, they use to be focused on partial treatments as the use of fluid resuscitation. A need of simulators based on flexible, incremental, upgradable and user understandable technologies is detected. For clinical and educational simulation in ICU's, these tools should also allow the simulation of the most prevalent shocks.

Decision tables are computer knowledge-based structures that could satisfy all these requirements. Our experience in the construction of a simulator for seven different shocks involving fluid resuscitation and vasoactive agent treatments concludes that the structure of the internal modified decision tables facilitates the knowledge engineer to identify the right questions for the physicians (or domain experts) during the knowledge acquisition process. Our proposed decision table structure also allows an easy representation of the acquired knowledge and the incremental versioning of knowledge about shocks. Extending the simulator with other shocks or diseases is also possible with the addition of new tables. Incorporating new clinical actions for a shock is also possible.

During this work, we observed that our simulation model with decision tables had two issues that required further consideration: On the one hand, decision tables provide similar response to patients who are in a similar state and receive the same treatment. In order to adjust this behavior to real patients arriving to an ICU who may evolve differently even if the same treatment is applied, we extended the simulator with a patient module (PM). Modeling patient's sensitivities/resistances to clinical actions, and patient's vital signs normality, was a complex process that concluded with an intuitive way to customize standard evolutions. The patient models are used by the PM in order to allow that different patients could evolve differently even if the same treatment is provided.

On the other hand, we consider that the patient's age, weight, and sex are closely related to the patient resistance to drugs and also to the response time of these drugs. In this work we have not included these relationships in the current simulator but they will be considered in future versions.

According to Gaba et al. [23], the purposes of simulation in health care can be classified in: *Education*, *Training*, *Performance assessment*, *Clinical Rehearsals*, and *Research*.

These are interesting areas of application of our Patient Simulation Model. In the immediate future we are addressing a couple of actions. On the one hand, we are defining with the Hospital Clínic de Barcelona a validation study of the model. This study will integrate several senior physicians of the ICU of this hospital that will interact with the model in order to provide feed-back on the correct and incorrect simulation allowing us to refine the model with new

versions of the decision tables contained. On the other hand, we will start a training program of the residents in the Emergency Unit of that same hospital. We are currently working in a on-line tool for residents to train their treatment recommendations for shocks with patient cases whose evolution will be calculated with our simulator as the users decide new clinical actions.

References

1. Vincent, J.L., Backer, D.: Circulatory shock. N. Engl. J. Med. **369**, 1726–1734 (2013)
2. Arturson, G., Groth, T., Hedlund, A., Zaar, B.: Computer simulation of fluid resuscitation in trauma. First pragmatic validation in thermal injury. J. Burn Care Rehabil. **10**(4), 292–299 (1989)
3. Roa, L., Gómez-Cía, T.: A burn patient resuscitation therapy designed by computer simulation (BET). Part 1: simulation studies. Burns **19**(4), 324–331 (1993)
4. Gómez-Cía, T., Roa, L.: A burn patient resuscitation therapy designed by computer simulation (BET). Part 2: initial clinical validation. Burns **19**(4), 332–338 (1993)
5. Dagan, I., Gabay, M., Barnea, O.: Fluid resuscitation: computer simulation and animal experiments. In: Conference Proceedings of IEEE Engineering in Medicine and Biology Society, pp. 2992–2995 (2007)
6. Sheffer, N., Hirshberg, A., Barnea, O.: Myocardial O2 balance during fluid resuscitation in uncontrolled hemorrhage: computer model. J. Trauma **42**(4), 647–651 (1997)
7. Magosso, E., Biavati, V., Ursino, M.: Role of the baroreflex in cardiovascular instability: a modeling study. Cardiovasc. Eng. Int. J. **1**(2), 101–115 (2001)
8. King, P.J.H.: Decision tables. Comput. J. **10**, 135–142 (1967)
9. Shiffman, R.N.: Representation of clinical practice guidelines in conventional and augmented decision tables. J. Am. Med. Inform. Assoc. **4**(5), 382–393 (1997)
10. Reynolds, H.R., Hochman, J.S.: Cardiogenic shock: current concepts and improved outcomes. Circulation **117**, 686–697 (2008)
11. Simons, F.E.R., Ardusso, L.R.F., Bilò, M.B., Dimov, V., Ebisawa, M., El-Gamal, Y.M., Ledford, D.K., Lockey, R.F., Ring, J., Sánchez-Borges, M., Senna, G.E., Sheikh, A., Thong, B.Y., Worm, M.: 2012 Update: world allergy organization guidelines for the assessment and management of anaphylaxis. Curr. Opin. Allergy Clin. Immunol. **12**, 389–399 (2012)
12. Konstantinides, S.V., Torbicki, A., Agnelli, G., et al.: 2014 ESC guidelines on the diagnosis and management of acute pulmonary embolism. Eur. Heart J. **35**, 3033–3080 (2014)
13. Maisch, B., Seferović, P.M., Ristić, A.D., et al.: Guidelines on the diagnosis and management of pericardial diseases. Executive summary. Eur. Heart J. **25**, 587–610 (2004)
14. Pascoe, S., Lynch, J.: Adult Trauma Clinical Practice Guidelines. Management of Hypovolaemic Shock in the Trauma Patient, NSW Institute of Trauma and Injury Management (2007)
15. Levi, L., Wolf, A., Belzberg, H.: Hemodynamic parameters in patients with acute cervical cord trauma: description, intervention, and prediction of outcome. Neurosurgery **33**(6), 1007–1017 (1993)
16. Consortium for Spinal Cord Medicine: Early Acute Management in Adults with Spinal Cord Injury: a clinical practice guideline for health-care professionals. J. Spinal Cord Med. **31**(4), 408–479 (2008)

17. Rivers, E., Nguyen, B., Havstad, S., et al.: Early goal-directed therapy in the treatment of severe sepsis and septic shock. New Engl. J. Med. **345**(19), 1368–1377 (2001)
18. Dellinger, R.P., Levy, M.M., Rhodes, A., et al.: Surviving sepsis campaign: international guidelines for management of severe sepsis and septic shock: 2012. Intensive Care Med. **39**(2), 165–228 (2013)
19. Shiffman, R.N., Greenes, R.A.: Rule set reduction using augmented decision table and semantic subsumption techniques: application to cholesterol guidelines. Proc. Annu. Symp. Comput. Appl. Med. Care, 339–343 (1992)
20. Schaafsma, M., van der Deijl, W., Smits, J.M., Rahmel, A.O., de Vries Robbé, P.F., Hoitsma, A.J.: Decision tables and rule engines in organ allocation systems for optimal transparency and flexibility. Transpl. Int. **24**(5), 433–440 (2011)
21. Riaño, D.: A systematic analysis of medical decisions: how to store knowledge and experience in decision tables. In: Riaño, D., ten Teije, A., Miksch, S. (eds.) KR4HC 2011. LNCS, vol. 6924, pp. 23–36. Springer, Heidelberg (2012)
22. Real, F., Riaño, D., Ramón Alonso, J.: Training residents in the application of clinical guidelines for differential diagnosis of the most frequent causes of arterial hypertension with decision tables. In: Miksch, S., Riaño, D., Teije, A. (eds.) KR4HC 2014. LNCS, vol. 8903, pp. 147–159. Springer, Heidelberg (2014)
23. Gaba, D.M.: The future vision of simulation in healthcare. Simul. Healthc. **2**(2), 126–135 (2007). doi:10.1097/01.SIH.0000258411.38212.32. Summer
24. Benneyan, J.C.: An introduction to using computer simulation in healthcare: patient wait case study. J. Soc. Health Syst. **5**(3), 1–15 (1997)
25. Barjis, J.: Healthcare simulation and its potential areas and future trends. SCS M&S Magazine **2**(5), 1–6 (2011)
26. Arturson, G., Groth, T., Hedlund, A., Zaar, B.: Potential use of computer simulation in treatment of burns with special regard to oedema formation. Comput. J. **10**, 135–142 (1967)

Clinical Guideline and Clinical Pathway Support

META-GLARE: A Meta-Engine for Executing Computer Interpretable Guidelines

Alessio Bottrighi[1(✉)], Stefania Rubrichi[1,2], and Paolo Terenziani[1]

[1] Computer Science Institute, DISIT, Univ. Piemonte Orientale, Alessandria, Italy
{alessio.bottrighi, stefania.rubrichi, paolo.terenziani}@uniupo.it
[2] Laboratory for Biomedical Informatics "Mario Stefanelli", Dipartimento di Ingegneria Industriale e dell'Informazione, University of Pavia, Pavia, Italy

Abstract. Clinical practice guidelines (CPGs) play an important role in medical practice, and computerized support to CPGs is now one of the most central areas of research in Artificial Intelligence in medicine. In recent years, many groups have developed different computer-assisted management systems of Computer Interpretable Guidelines (CIGs). We propose a generalization: META-GLARE is a "meta"-system (or, in other words, a shell) to define new CIG systems. It takes as input a representation formalism for CIGs, and automatically provides acquisition, consultation and execution engines for it. Our meta-approach has several advantages, such as generality and, above all, flexibility and extendibility. While the meta-engine for acquisition has been already described, in this paper we focus on the execution (meta-)engine.

Keywords: Computer interpretable guideline (CIG) · Metamodeling for healthcare systems · Meta CIG system · System architecture · CIG execution

1 Introduction

Clinical practice guidelines (CPGs) represent the current understanding of the best clinical practice. In recent years the importance and the use of CPGs are increasing in order to improve the quality and to reduce the cost of health care. ICT technology can further enhance the impact of CPGs. Thus, in the last twenty years, many different systems and projects have therefore been developed in order to manage computer interpretable CPGs. A survey and/or a comparative analysis of these systems are outside the goals of this paper. A comparison of Asbru, EON, GLIF, Guide, PROforma, PRODIGY can be found in [1]. In [2] the comparison has been extended to GLARE and GPROVE. The books [3, 4] represent a quite recent consensus of a part of the computer-oriented CIG community. A recent and comprehensive survey of the state-of-the-art about CIG has been published by Peleg [5].

The surveys/books show that few important commonalities have been reached. In particular, most approaches model guidelines in terms of a Task-Network Model (TNM): a (hierarchical) model of the guideline control flow as a network (graph) of specific tasks

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 37–50, 2015.
DOI: 10.1007/978-3-319-26585-8_3

(represented by nodes). Although the terminology may differ, all approaches support a basic set of core guideline tasks, such as decisions, actions and entry criteria. From the architecture point of view, most CIG approaches provide specific support for at least two subtasks: (i) CIG acquisition and representation and (ii) CIG execution, providing engines which support the execution of an acquired CIG on a specific patient. However, there are also important distinguishing features between the different CIG systems, often due to the fact that many of such systems are mostly research tools that evolve and expand to cover an increasing number of phenomena/tasks.

1.1 Origin and Motivation of the META-GLARE Approach

Such an evolution characterizes the history of GLARE (Guideline Acquisition, Representation, and Execution) [6, 7], the prototypical system we have been building since 1996 in cooperation with ASU San Giovanni Battista in Turin, one of the major hospitals in Italy. In our experience, it is quite frequent that, due to the need of facing a new real-world clinical guideline domain, some extensions to a CIG system are needed. In complex CIG systems (like GLARE) extensions require a quite large amount of work, since different parts of the code system must be modified, and their interactions considered. Specifically, extensions to the representation formalisms always involve the need of modifying the code of the acquisition, the consultation, and the execution engines of the system. On the other hand, the possibility of easily and quickly extend systems, and, more generally, of achieving ***fast prototyping*** when approaching new domains and\or tasks are essential in this field of AI in medicine research.

With such goals in mind, we started to re-design GLARE. Initially, we wanted to design yet a new CIG system, based on a new CIG formalism, enclosing the "best features" of current CIG approaches in the literature. However:

(1) such a general formalism would certainly be very general, and complex. However, CIGs have to be managed by physicians, so that simplicity (also in terms of the number of representation primitives being proposed) is a strict requirement. (2) the long-term experience of AI research has definitely shown that there is no "perfect" formalism. Whatever general CIG formalism could be defined, for sure can still require extensions, when facing new phenomena.

As a consequence of (1) and (2), we decided to pursue a completely different and innovative goal: instead of defining "yet another new CIG formalism and system", we chose to devise a "meta-system" (called META-GLARE), or, in other words, a shell supporting the definition of new CIG formalisms and systems (or facilitating the extensions of them). Though this idea is entirely new in the CIG literature, it stems from software engineering consolidated methodologies, and from the recent meta-modeling field in the medical informatics (see Sect. 6).

Our meta-system, called META-GLARE

(i) makes "minimal" assumptions as regards the CIG formalisms (basically, it ***simply assumes that CIGs are represented through TNM***)

(ii) ***provides general acquisition, consultation and execution engines, that are parametric over the specific CIG formalism being considered*** (in other words, the CIG formalism is an input of such engines).

1.2 Methodology and Advantages of the META-GLARE Approach

The core idea of our meta-approach is

(i) To define an open library of elementary components (e.g., textual attribute, Boolean condition, Score-based decision), each of which was equipped with methods for acquiring, consulting and executing it
(ii) To provide system-designers with an easy way of defining node and arc types (constituting the representation formalism of a new system) in terms of the elementary components constituting them
(iii) To devise general and basic tools for the acquisition, consultation and execution of CIGs, which are parametric with respect to the formalism used to represent them (in the sense that the definition of node and arc types are an input for such tools).

In such a way, we achieve several advantages:

- The definition of a new system (based on a new representation formalism) is easy and quick. Using META-GLARE, a system designer can easily define her/his own new system, by defining its formalism: (i) the node types, (ii) the arc types (both are defined types as an aggregation of components from the library), and (possibly) the constraints on them. No other effort (e.g., building acquisition or execution modules) is required.
- The ***extensions*** to an existing system (through the modification of its representation formalism) are easy and quick. In META-GLARE, a system designer can extend a system by defining and adding new node/arc types, or adding components to existing types (with no programming effort at all)
- ***User programming is needed only in case a new component has to be added in the component library***. However, the addition is ***modular*** and minimal: the system designer has just to focus on the added component, and to provide the code for acquiring, consulting, and (if needed) execute it. Such programming is completely "local" and "modular": the new methods have to be programmed "in isolation" (without having to care of the rest of the system). *No modification to any software component in META-GLARE architecture* (see Fig. 1 below) *is required to integrate the new methods*: META-GLARE automatically evokes them when needed during acquisition, consultation and execution.
- As a consequence, fast prototyping of the new (or extended) system is achieved (see the examples in Sect. 5).

We have already presented our innovative idea of proposing a "shell" for designing new CIG systems, and META-GLARE architecture, in the previous KR4HC workshop [8]. In such a paper, we have also described META-GLARE acquisition engine. This

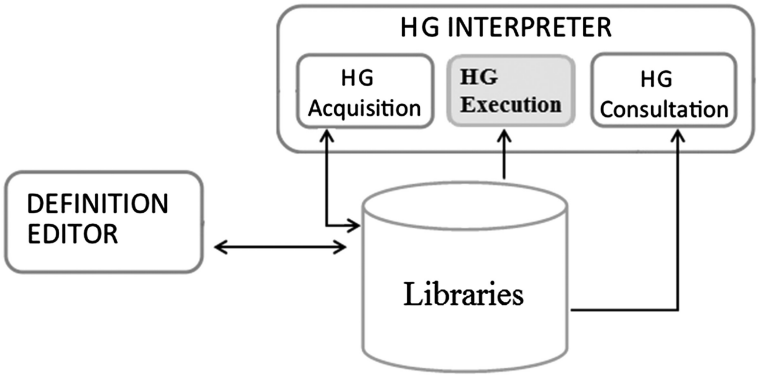


Fig. 1. The architecture of META-GLARE

paper is the natural completion of such a previous work, with the description of META-GLARE execution engine.

2 META-GLARE Architecture

META-GLARE supports any CIG representation formalism based on the following aspects: **(1)** guidelines are represented by **hierarchical graphs** (**constraints** on the graph –e.g., acyclicity- also supported); **(2)** there is no assumption on which **types** of **nodes** and **arcs** can be used to describe the graphs. The only assumption is that each **type** (of node and of arc) is defined as **a list of attributes**. **(3)** There is no assumption on which attribute types may be introduced in a specific formalism. We distinguish between two main categories of attributes: **control** attributes (i.e., those attributes that affect the execution of a node/arc; e.g., decision attributes) and **non-control** ones (e.g., textual attributes).

Thus, META-GLARE interpreter (guideline acquisition, consultation and execution tools) only assumes that a guideline is a hierarchical graph, and is ***parametrized*** on the **types of nodes** and **arcs,** and on the **types of attributes.** The interpretation of each node/arc type is obtained compositionally through the sequenced interpretation of the attributes composing it. This means that, practically, *each attribute type* (e.g., textual attribute, Boolean condition attribute, etc.) *must consists of the methods to acquire, consult, and (possibly) execute it*. Thus, for instance, guideline acquisition consists in the acquisition of a hierarchical graph, which in turn adopts the methods in each attribute type definition to acquire the specific attributes of the involved nodes/arcs.

In Fig. 1, we show a simplified version of the architecture of META-GLARE, focusing on parts affecting execution (for a more extensive description, see [8]). Oval nodes represent data structures, and rectangles represent computational modules. The DEFINITION_EDITOR tool supports system-designers in the definition of a new

system. It consists in four sub-components, to cope with the definition of (i) attribute types, (ii) node/arc types, (iii) graph constraints, and (iv) CIG formalism (where a CIG formalism is just a set of node/arc types and (possibly) of graph constraints). The output is an XML representation in the library.

Globally, the DEFINITION_EDITOR module manages the definition of the formalism a new CIG system. On the other hand, the HG_INTERPRETER (HG stands for "hierarchical graph") deals with the aspects which common to all the systems that can be generated by META-GLARE. It consists of three sub-components: HG_ACQUISITION, HG_CONSULTATION, and HG_EXECUTION. META-GLARE and its modules are developed as *Java Applets*. In this way, META-GLARE is a cross-platform application: it can be embedded into a web page and executed via web browsers without any installation phase. The libraries in Fig. 1 are implemented by databases stored in PostgreSQL. In the paper, we focus on the HG_EXECUTION module only.

3 CIG Execution Meta-Engine

The HG_execution module takes as input:

(1) A formalism (i.e., a set of arc/nodes types, each one consisting of a sequence of attributes)
(2) A specific CIG (the one to be executed), expressed in the given formalism
(3) A specific patient (whose data are collected in a database).

HG_execution supports the execution of the CIG on the specific data. Notice that, while all the execution engines in the CIG literature supports are specifically designed for the execution of a specific CIG formalism (so that their input are only (2) and (3) above), here the executor much more general, since any input formalism must be executable (i.e., also (1) is an input for the (meta-) executor). Our (meta-) executor only assumes that a guideline is a hierarchical graph, and is ***parametrized*** on the **types of nodes** and **arcs,** and on the **types of attributes.** As a consequence, it "inherits" from the CIG execution engines in the literature (see [9]) the way they deal with hierarchical graphs (points (i), (ii), and (iii.a) below), but it is parametric on the methods used to execute control attributes (point (iii.b) below). The basic idea in the definition of the (meta-) executor is simple:

(i) The execution of a CIG is the sequential execution of its nodes
(ii) Each node in the CIG is an instance of a type of node in the input formalism, and the node definition basically consists of a sequence of typed attributes
(iii) If the attribute is not a control attribute, (iii.a) it can be ignored by the executor. Otherwise, (iii.b) each (type of) control attribute has a method stating how it has to be executed. The executor simply execute such a method.

However, many refinements are required, concerning point (i) above (such refinements are considered also by most CIG engines in the literature). Basically, three

main problems have to be addressed: (1) the graph is hierarchical, (2) the graph is not simply a sequence of nodes (different types of arcs may be used; e.g., arcs for specifying alternatives, or concurrence), and (3) the flow of control may be altered by the execution of (the execution method of) a control attribute (e.g., the conditioned_GOTO or REPEAT attributes – see Sect. 3).

Regarding issue (1), since we assume that graphs may be hierarchical, we support the treatment of composite nodes, which are defined in terms of their components (which, in turn, are hierarchical graphs). The execution of a composite node starts with the execution of the first node of the sub-graph defining it, and ends when the execution of such a subgraph ends. Thus the nesting of calls to subgraphs must be explicitly managed by the executor.

As regards (2), point (i) naively assumes that graphs are defined only using one type of arc, representing the sequence in which two nodes have to be executed. However, notice that also the (types of) arcs are part of the definition of the input formalism. Thus, our engine must support the treatment of user-defined arcs, where each arc type is defined as a sequence of attributes. If an arc does not contain any control attribute, it can simply be ignored by the executor. On the other hand, if it has a control attribute, its method must be executed, to determine which is the next node to be executed. For the sake of brevity, in this paper we consider only directed arcs (as all CIG approaches do, to the best of our knowledge). We admit arcs with one starting node but multiple ending nodes (to support alternatives, concurrence, etc.). Thus, a set of nodes (to be executed) may be the result of the execution of an arc. Thus, also the fact that multiple actions may be candidate to be executed next must be managed by the executor.

As regards point (3), different types of control attributes can be used in a formalism. Our current library of attributes is briefly described in Sect. 3, and includes the main types provided by the CIG approaches in the literature [9]. However, we stress that such a library is *open*: new attribute types can be introduced when a new formalism (or an extension to a current formalism) is defined and no modification of HG_INTERPRETER is needed to manage them. Indeed, in many cases, such attributes can determine an "alteration" of the flow of control represented through the arcs in the hierarchical graph. For instance, an attribute type modeling repetition (REPEAT in our current library) can state that the next node to be executed is the current node itself; an attribute type modeling conditional GOTO (ConditionedGOTO in our current library) can state that the next node to be executed is the another node in the CIG, and so on. These alteration of the "standard" control flow of the graph must be managed, too, by the executor.

In order to cope with issues (1) and (2) above, the executor adopts a data structure (the ***execution_tree***) to explicitly represent, at each step, the hierarchy of active composite nodes (which are represented by the internal nodes of the tree) and the set of atomic nodes which are candidate to be executed next (the are represented by the leaves of the tree). The root of the tree is (by default) a "dummy" node representing the whole CIG, and each node in the tree is a "pointer" to a node in the CIG being executed.

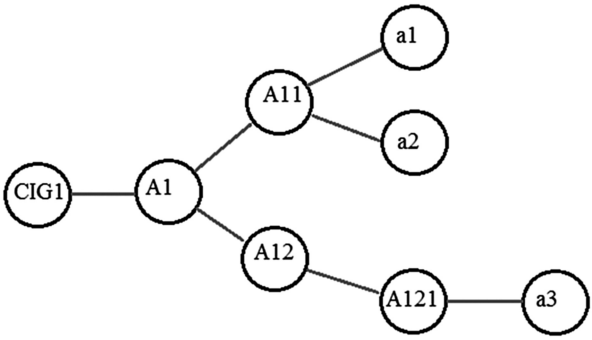


Fig. 2. Execution_tree: an example.

Of course, the execution_tree must be properly updated by the executor after the execution of each atomic node.

For example, the tree in Fig. 2 represents a situation in which CIG1 is being executed. The current composite action being executed is A1. The composite nodes A11 and A12 are the components of A1 currently being executed (concurrently). In turn, the currently executable (atomic) nodes in A11 are the atomic nodes a1 and a2. On the other hand, the active node composing A12 is still a composite node (A121), and its active component is the atomic node a3.

Concerning issue 3, though the library of attributes is open, we impose the constraint that each execution method in the library must return to the executor an indication of whether and how they affect the control flow. Six cases are considered (covering, to the best of our knowledge, the main possibilities considered by CIG execution engines in the literature [9]; see the discussion in Sect. 6):

(1) "**go_on**": this is the standard continuation. With a "go_on", the executor must execute the next control attribute in the node (or has finished the execution of the node, if the current attribute is the last control one) (Algo 1 line 19)
(2) "**repeat**": the current control attribute must be executed again (Algo 1 lines 9–11)
(3) "**suspend**": the execution of the current CIG has to be suspended (Algo 1 lines 12–13)
(4) "**abort**": the execution of the current CIG has to be stopped, and terminates. (Algo1 line 16)
(5) "**goto**" <node>: the execution of the CIG restarts from the execution of <node> (Algo 1 line 18)
(6) "**fail**": a failure has occurred (e.g., an action could't be executed, because the required instrument is not available). The current execution is stopped, and the executor must enact the general recovering facility. (Algo 1 line 17).

The execution of a CIG starts with the initialization of the execution tree. Then, the executor operates as abstractly described by the algorithm Algo 1.

```
1.  Executor_algo(nodes/arc types definitions, library of attribute types, CIG, patient
    data, Et: execution tree)
2.  while (not null Et) do
3.  begin
4.          Cur_n ← choose one leaf from Et;
5.          Modality ← "go_on";
6.          while ((there are control attributes in Cur_n) AND (modality = "go_on")) do
7.          begin
8.                  Modality←exec(control_attribute.execmethod)
9.                  if      (Modality = "repeat") then
10.                         repeat Modality← exec(control_attribute.execmethod)
11.                         until Modality≠"repeat";
12.                 if      (Modality = "suspend") then
13.                         wait until modality = "go_on"
14.         end;
15.         case Modality of
16.                 "abort": delete(Et) Exit;
17.                  "fail": manage failure;
18.                  "goto" <node>: Et←construct_new_tree(<node>,CIG);
19.                 "go_on": update_tree(Cur_n, Et, CIG);
20.         endcase;
21. end.
```

Algorithm 1. Pseudo-code of the algorithm for the CIG execution

Leaves of the execution_tree represent possibly concurrent actions, so that it is up to the user-physician to select which leaf has to be executed next (Algo 1 line 4). In the case of "goto" (Algo 1 line 18), the current execution tree is substituted by a new tree, in which a "pointer" to <node> is a leaf. However, also the "upper parts" of such a tree must be built (considering the composite actions in the CIG containing <node>, if any), by *construct_new_tree*. The "standard" continuation ("go_on" modality, Algo 1 line 19) is managed through a proper update of the execution tree, as shown by Algo 2 in the Appendix.

4 Control Attributes

In META-GLARE, two basic categories of control attribute types can be defined: the control attributes of nodes, and the one of arcs.

4.1 Control Attributes of Nodes

In our approach, attribute types are characterized by several *features*, which are specified according to the XML document. A XML tag, which describes an *attribute type*, has several *features*, defining its name, its properties, and its interpretation. "Interpretation" tags are very important, since they define (pointers to) the methods that are used by the HG_INTERPRETER to acquire, store, consult and execute any instance of such types.

Notably, the XML definition of "procedural" attributes does not contain the Java code of the methods, but only symbolic pointers to them.

We have currently several control attributes types. However, the library is open, and new attributes can be added by users. Indeed, the only constraint is that the execution methods must return as output one of the five modalities discussed in Sect. 3. At the current stage, we have implemented: BooleanCondition, BooleanDecision, ScoredCondition, ScoredDecision, DataEnquiry, FAIL, ConditionedFAIL, ABORT, ConditionedABORT, GOTO, ConditionedGOTO, REPEAT, external_action. Names are mostly self-explicative. Conditions are evaluated on the basis of patient's data automatically retrieved from a database. Decisions (between alternatives) are evaluated by evaluating the condition associated with each one of the alternatives, and showing to the user the result of the automatic evaluation. The final decision between alternatives is left to the physicians. DataEnquity is a data request. Data are retrieved from the database. If they are not present, the execution engine waits for them. External_action is a special control attribute type, to model those activities that have to be performed (e.g., by physicians, nurses, etc.) on the patient (e.g., the administration of a drug). The corresponding methods simply ask to the user whether the action has been successfully performed (returning the "go_on" modality), or not (returning the "fail" modality).

4.2 Control Attributes of Arcs

As discussed in Sect. 3, we support arcs having one starting node, and one or many ending nodes. Each arc is described by a set of attributes. Control arcs have exactly one control attribute (but they may have other non-control ones). The library is open, and currently contains the definition of three control attributes for arcs: ***sequence***, ***alternative***, and ***concurrence***. These three types cover the most significant cases considered in the literature, except "constrained" arcs in GLARE, which support complex temporal constraints between nodes. Sequence has just one ending node, and its execution method simply returns it. Concurrence has two or more ending nodes, and returns them. Alternative has two or more ending nodes, and asks to the user-physician to choose one of them (which is returned).

5 META-GLARE in Action

META-GLARE takes as input a CIG formalism (based on the TNM model) and provides as output a CIG system to acquire and execute guidelines represented on top of such a formalism. Thus, it support fast CIG system prototyping, both when building a new CIG system (on the basis of a new TNM-based formalism), and when extending a current one (with the addition of new features to a CIG formalism).

Example 1. As a first concrete example, suppose that, while analysing a new domain, a system designer identifies the need of enriching her\his CIG formalism (or to design a new CIG formalism) with a new type of node, consisting of

- some non-control attributes (e.g., textual attributes for name and goal, numerical attribute for cost, and so on)
- a sequence of three control attributes, "precondition" (of type ConditionalFAIL), "body" (of type external_action) and "postcondition" (of type ConditionalFAIL).

In particular, the intended purpose of (the control attributes of) the node is that, during execution, preconditions (which are Boolean conditions) are first checked on the patient data. If they are not satisfied, the execution of the node fails (and fail recovery is started); otherwise, an external action is executed on the patient. After the execution, post-conditions are verified, possibly leading to a failure of the execution of the node.

All the required attribute types in the example are already present in META-GLARE Library (textual and numerical types for non-control attributes; type ConditionFAIL for "preconditions" and "postconditions"; type "external_action" for the external action). As a consequence, the system-designer has simply to enter the new node definition (through the graphical interface of the DEFINITION_EDITOR). No other effort is required. The whole work requires only few minutes to the system-designer. As a result, the executor described in Sect. 3 will deal also with CIGs having such a new type of nodes, without requiring any modification.

On the other hand, some programming is required in case a new attribute type (not already existent in the library) has to be added.

Example 2. When choosing among clinically "equivalent" (with respect to the patient at hand) therapies, the "long term" effects of the therapeutic choice (e.g., what path of actions should be performed next, what are their costs, durations and expected utilities) may be helpful to discriminate. Decision theory [10] may be helpful in this context. In particular, it allows to identify the optimal policy, i.e., the (sequence of) action(s) changing the states of the system in such a manner that a given criterion is optimized. In order to compare the different action outcomes, one commonly assigns a utility to each of the reached states. Since there is uncertainty in what the states will be, the optimal policy maximizes the expected utility. It has been recently shown how decision theory can be exploited in the CIG context, to model *cost/benefit decisions* [11]. Currently, no facility for cost/benefit decisions is provided in META-GLARE Library. Thus, to extend a (META-GLARE-based) CIG formalism with a new type of control attribute (say Cost/Benefit_Decision) modelling cost/benefit decisions, new methods must be developed by the system designer, to acquire, consult, and execute it. But ***no*** modification to META-GLARE acquisition, consultation and execution engines has to be performed. In particular, focusing on execution, it is fundamental to stress that, as long as the new execution method Cost/Benefit_Decision returns one of the six modalities managed by the executor, no modification to the executor itself is needed at all.

6 Related Works, Conclusions, and Future Work

In this paper, we propose a (partial) description of the execution engine of META-GLARE, an innovative approach to cope with CIGs.

The HG_INTERPRETER, discussed in Sect. 3 is general, in that it covers a family of different formalisms (all the formalism that can be defined by META-GLARE). In such a way, it is neatly different from all the others CIG interpreters in the literature, which are biased to the treatment of a specific formalism [9]. In particular, different CIG interpreters have been compared in [9], where it is highlighted that, with the only exception of PROforma [12], that uses Prolog interpreter for execution, all the other systems have developed their own formalism-dependent interpreter. However, some of them share common features. Systems such as Arezzo^{TM} [13], GLARE [7], HeCaSe2 [14], SAGE [15] and GLEE [16] use similar basic elements (actions, decisions and enquiries). In particular, to enhance the generality of our approach, the META-GLARE interpreter covers (to the best of our knowledge) all the main modalities considered by the different interpreters in the literature. Notably, our treatment of modality partly encompasses the state transition model for the execution of a single action of approaches like PROforma [12] and Asbru [17] (which, for instance, consider states like "abort" or "suspend"; other states, like "in progress" or "completed" represent the "standard" execution of an action, so that they are implicitly managed by our meta-interpreter).

The main idea underlying META-GLARE is simple: instead of proposing "yet another system" to acquire, represent and execute CIGs, we propose a "meta-system", i.e., a shell to define (or modify) CIG systems. Roughly speaking, the input of META-GLARE is a description of a representation formalism for CIGs, and the output is a new system able to acquire, represent, consult and execute CIGs described using such a formalism. Indeed, such a basic idea is not at all new in Computer Science, although it is the first time that it has been applied to the domain of CIGs.

A similar idea, in a completely different context, has emerged in Computer Science in the 70', with the definition of the so-called "compilers of compilers", like YACC (Yet Another Compiler of Compilers [18]). In particular, META-GLARE takes as input any CIG formalism and provides as output a CIG system (i.e., an acquisition, a consultation and an execution engine) for such formalism just as YACC takes as input any context free language (expressed through a formal attribute grammar) and provides as output a compiler for it. More recently, Model-Driven Software Engineering (MDSE) has emerged as a promising methodology for software systems, targeting challenges in software engineering relating to productivity, flexibility and reliability. MDSE is especially useful as a methodology for the development of healthcare systems, and even a dedicated workshop (the International Workshop on Metamodelling for Healthcare Systems, 2014 (http://mmhs.hib.no/2014/) and 2015 (http://mmhs.hib.no/2015/) has been created to face such a topic).

Indeed, the application of models to software development is a long-standing tradition, and has become even more popular since the development of the Unified Modeling Language (UML). Yet we are faced with 'mere' documentation, MDSE has an entirely different approach: Models do not constitute documentation, but are considered equal to code, as their implementation is (semi)automated. MDSE therefore aims to find domain-specific abstractions and makes them accessible through formal modeling. This procedure creates a great potential for automation of software production, which in turn leads to increased productivity, increasing both the quality and maintainability of software systems. We share part of the methodology of MDSE, such as the use of three levels of models (the meta-formalism level, the formalism level, and

the CIG instance level), but a relevant difference should be pointed out. In "standard" MDSE approaches, the final model is used to semi-automatically generate the application code, through the adoption of transformation rules. On the other hand, there is no semi-automatic code generation in META-GLARE. Indeed, the HG-interpreter is already provided by META-GLARE, but it is parametrized over the CIG formalism, so that a CIG interpreter is automatically obtained when a specific CIG formalism is selected.

To the best of our knowledge, the application of such ideas to the context of CIG is completely new. Such an application has mainly motivated by our goal of designing and implementing a flexible and powerful vehicle for research about CIG. In our opinion, META-GLARE provides two main types of advantages, both strictly related to the notion of *easy* and *fast prototyping*. Using META-GLARE

(1) the definition of a new system (based on a new representation formalism) is easy and quick;
(2) The extension of an existing system (through the modification of the representation formalism) is easy and quick.

In particular, the executor described in Sect. 3 will deal also with CIGs having such a new type of nodes, without requiring any modification. On the other hand, some programming is required in case a new attribute type (not already existent in the library) has to be added (see discussion in Sect. 5).

Thus, META-GLARE is, above all, a good vehicle for fast definition/extension and prototyping CIG systems, making it quite suitable especially as a research tool to address new CIG phenomena.

The implementation of META-GLARE execution engine is actually ongoing. We plan to finish it as soon as possible, to start with an extensive experimental evaluation of our approach. Though quite powerful, the current approach has several limitations, which we want to overcome in our future work. In particular, we want to consider the addition of new modalities (besides the ones discussed in Sect. 3), and we aim to extend our current approach to deal (i) with exceptions (along the lines discussed in [19] and (ii) with the concurrent (but possibly interacting) execution of two or more CIGs, to cope with comorbidities (integrating the work in [20] into META-GLARE)).

Acknowledgements. The research described in this paper has been partially supported by Compagnia San Paolo, within the GINSENG project.

Appendix

Algorithm Algo 2 in the following describes how to update of the execution tree, in case of "go_on" modality.

```
1.  update_tree(Node,Tree,CIG)
2.  begin
3.          Mother_n←get_mother(Node,Tree);
4.          delete(Node,Tree);
5.          if Node has brothers then return Tree;
6.          else
7.                  begin
8.                  Next_nodes←get_next(Node,CIG);
9.                  if (Next_nodes ≠ null) then
10.                         for each Node∈ Next_nodes
11.                         do append(expand_down(Node,CIG), Mother_n);
12.                         return Tree;
13.                 else return update_tree(Mother_n, Tree,CIG)
14.                 end
15. end.
```

Algorithm 2. Pseudo-code of the algorithm for the update of the execution tree, in case of "go_on" modality.

Once a node has been executed, it is deleted from the execution tree. In case there are concurrent nodes to be executed (brothers) (line 5), the executor simply has to operate on such a new tree. Otherwise (lines 7–13), the deleted node has to be substituted by the immediately-next nodes to be executed in the CIG (in the case of concurrent actions, there is more then one "immediatly-next" node to be considered). The function ***get_next*** consider the control arc (which must be unique, if it exist) exiting from Node in the CIG, and execute it is execution method (line 8). As a result, a set of next nodes to be executed is returned. Each one of such nodes must be added to the tree (***append*** function), and possibly expanded (***expand_down***: if Node is composite, then the first nodes (in the case of concurrent actions, there is more then one "first" node to be considered) of the CIG subgraph representing it are appended to treeNode, and so, on, recursively, until atomic nodes are reached, lines 10–12). On the other hand (line 13), if there are no next node (i.e., if the executed node was the last one in a graph or subgraph), then the update_tree algorithm must be recursively applied on the mother of the current node.

References

1. Peleg, M., Tu, S., Bury, J., Ciccarese, P., Fox, J., Greenes, R.A., Hall, R., Johnson, P.D., Jones, N., Kumar, A., Miksch, S., Quaglini, S., Seyfang, A., Shortliffe, E.H., Stefanelli, M.: Comparing computer-interpretable guideline models: a case-study approach. JAMIA **10**(1), 52–68 (2003)
2. Bottrighi, A., Chesani, F., Mello, P., Montali, M., Montani, S., Storari, S., Terenziani, P.: Analysis of the GLARE and GPROVE approaches to clinical guidelines. In: Riaño, D., ten Teije, A., Miksch, S., Peleg, M. (eds.) KR4HC 2009. LNCS, vol. 5943, pp. 76–87. Springer, Heidelberg (2010)

3. ten Teije, A., Miksch, S., Lucas, P. (eds.): Computer-Based Medical Guidelines and Protocols: a Primer and Current Trends. IOS Press, Amsterdam (2008)
4. Lucas, P., Hommerson, A. (eds.): Foundations of Biomedical Knowledge Representation. Springer, Heidelberg (2015)
5. Peleg, M.: Computer-interpretable clinical guidelines: a methodological review. J. Biomed. Inform. **46**(4), 744–763 (2013)
6. Terenziani, P., Molino, G., Torchio, M.: A modular approach for representing and executing clinical guidelines. Artif. Intell. Med. **23**(3), 249–276 (2001)
7. Terenziani, P., Montani, S., Bottrighi, A., Molino, G., Torchio, M.: Applying artificial intelligence to clinical guidelines: the GLARE approach. In: [3], 273–282 (2008)
8. Terenziani, P., Bottrighi, A., Lovotti, I., Rubrichi, S.: META-GLARE: a meta-system for defining your own CIG system: architecture and acquisition. In: Miksch, S., Riano, D., ten Teije, A. (eds.) KR4HC 2014. LNCS, vol. 8903, pp. 95–110. Springer, Heidelberg (2014)
9. Isern, D., Moreno, A.: Computer-based execution of clinical guidelines: A review. Int. J. Med. Inform. **77**, 787–808 (2008)
10. Russel, S., Norving, P.: Artificial Intelligence: a Modern Approach. Prentice Hall, New Jersey (2009)
11. Anselma, L., Bottrighi, A., Molino, G., Montani, S., Terenziani, P., Torchio, M.: Supporting knowledge-based decision making in the medical context: the GLARE approach. IJKBO **1** (1), 42–60 (2011)
12. Sutton, D.R., Fox, J.: The syntax and semantics of the PROforma guideline modeling language. J. Am. Med. Inform. Assoc. **10**, 433–443 (2003)
13. InferMed, Arezzo Technical White Paper, Technical report InferMed, Ltd. http://www.infermed.com/ Accessed 18 May 2015
14. Isern, D., Sánchez, D., Moreno, A.: HeCaSe2: A Multi-agent Ontology-Driven Guideline Enactment Engine. In: Burkhard, H.-D., Lindemann, G., Verbrugge, R., Varga, L.Z. (eds.) CEEMAS 2007. LNCS (LNAI), vol. 4696, pp. 322–324. Springer, Heidelberg (2007)
15. Tu, S.W., Campbell, J.R., Glasgow, J., Nyman, M.A., McClure, R., McClay, J., Parker, C., Hrabak, K.M., Berg, D., Weida, T., Mansfield, J.G., Musen, M.A., Abarbanel, R.M.: The SAGE guideline model: achievements and overview. JAMIA **14**(5), 589–598 (2007)
16. Wang, D., Peleg, M., Tu, S.W., Boxwala, A.A., Ogunyemi, O., Zeng, Q., Greenes, R.A., Patel, V.L., Shortliffe, E.H.: Design and implementation of the GLIF3 guideline execution engine. J. Biomed. Inform. **37**, 305–318 (2004)
17. Young, O., Shahar, Y., Liel, Y., Lunenfeld, E., Bar, G., Shalom, E., Martins, S.B., Vaszar, L.T., Marom, T., Goldstein, M.K.: Runtime application of Hybrid-Asbru clinical guidelines. J. Biomed. Inform. **40**, 507–526 (2007)
18. Johnson, S.C.: Yacc: Yet Another Compiler-Compiler, vol. 32. Bell Laboratories, Murray Hill, NJ (1975)
19. Leonardi, G., Bottrighi, A., Galliani, G., Terenziani, P., Messina, A., Della Corte, F.: Exceptions handling within GLARE clinical guideline framework. AMIA Annu. Symp. Proc. **2012**, 512–521 (2012)
20. Piovesan, L., Molino, G., Terenziani, P.: Supporting Physicians in the Detection of the Interactions between Treatments of Co-Morbid Patients, In: Tavana, M., Ghapanchi, A.H., Talaei-Khoei A. (Eds.) Healthcare Informatics and Analytics: Emerging Issues and Trends, Chapter: 9, IGI Global (2014)

Identifying Evidence Quality for Updating Evidence-Based Medical Guidelines

Zhisheng Huang(✉), Qing Hu, Annette ten Teije, and Frank van Harmelen

Department of Computer Science,
VU University Amsterdam, Amsterdam, The Netherlands
{huang,qhu400,annette,Frank.van.Harmelen}@cs.vu.nl

Abstract. Evidence-based medical guidelines contain a collection of recommendations which have been created using the best clinical research findings (a.k.a. evidences) of the highest value to aid in the delivery of optimum clinical care to patients. In evidence-based medical guidelines, the conclusions (a.k.a. recommendations) are marked with different evidence levels according to quality of the supporting evidences. Finding new relevant and higher quality evidences is an important issue for supporting the process of updating medical guidelines. In this paper, we propose a method to automatically identify all evidence classes. Furthermore, the proposed approach has been implemented by a rule-based approach, in which the identification knowledge is formalized as a set of rules in the declarative logic programming language Prolog, so that the knowledge can be easily maintained, updated, and re-used. Our experiments show that the proposed method for identifying the evidence quality has a recall of 0.35 and a precision of 0.42. For the identification of A-class evidences (the top evidence class), the performance of the proposed method improves to recall = 0.63 and precision = 0.74.

1 Introduction

Evidence-based medical guidelines contain a collection of recommendations which have been created using the best clinical research findings (a.k.a. evidences) of the highest value to aid in the delivery of optimum clinical care to patients. Evidence-based medical guidelines (from now on simply "guidelines" for short) provide important knowledge sources for medical decision making. In such guidelines, recommendations are marked with different levels according to the quality classes of the supporting evidences.

Usually evidences are classified into the following five evidence classes: A1-class evidences are the top-class which are based on systematic reviews, A2-class evidences are based on high-quality randomized controlled trials, B-class evidences are based on randomized controlled trials of moderate quality, C-class evidences are those based on non-comparative trials, and D-class evidences are based on the personal opinion of experts.

A guideline should be regularly updated when new relevant evidences appear in the literature. The updated guideline can better serve medical practice by

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 51–64, 2015.
DOI: 10.1007/978-3-319-26585-8_4

using the latest medical research evidence [13] which can be retrieved from medical publication repositories like PubMed. Thus, an evidence can be identified with its PubMed ID (PMID). Automatically finding relevant evidences by computers has been considered to improve the efficiency of guideline update. Finding relevant and high quality of new evidences becomes one of the important issues for medical guideline update.

There have been various researchers working on finding relevant evidences for guideline update [6,10]. However, there has been little work on how to identify the quality of the evidence while searching for new relevant evidence. Cohen et al. describe a system that can identify systematic reviews (i.e., A1-class evidences) [1]. This approach uses a machine learning method based on Support Vector Machines that is trained on a dataset of pre-tagged articles. This led to good results in testing, where over 70 % of all updates were recognized while maintaining a low alert rate. Rosefeld and Shiffman describe a set of criteria that can be used to determine a strong evidence (i.e., a A-class evidence or some B-class evidence) [12]. Iruetaguena et al. implemented them and used them for filtering their search results [5]. Both Cohen's approach and Rosefeld and Shikman approach are too rough for the classification of evidence levels, because they do not cover all of the evidence classes. Thus, it is useful to develop an approach which can identify the evidence quality that covers all of the evidence classes.

In this paper, we propose a method to automatically identify all evidence classes by extending the meta-data checking which cover all evidence classes. Furthermore, this extension is implemented by using a rule-based approach, in which the identification knowledge is formalized as a set of rules in the declarative logic programming language Prolog. The main advantage of this rule-based approach is that the identification knowledge is formalized as a set of rules, rather than being encoded in programming codes, so that the knowledge can be easily maintained, updated, and re-used [3]. The existing algorithms (e.g.,Cohen's approach, and Rosefeld and Shiffman approach) can be easily integrated as a subset of rules with our implementation.

We have used the evidences in the 2004 and 2012 versions of the Dutch Breast Cancer Guideline as test data [8]. The evidences in those two guidelines have been well classified according to their evidence classes. Thus, those evidences can serve as a gold standard for the evaluation of the proposed methods. Our experiments show that the proposed method of identifying the evidence class level has a recall of 0.35 and a precision of 0.42. For the identification of A-class evidences (the top evidence class), the proposed approach can provide a better result with recall=0.63 and precision=0.74.

The main contributions of this paper are: (i) we propose an approach to evidence quality identification which covers all of the evidence classes, (ii) we propose a rule-based approach to formalize the knowledge required for identifying the quality level of evidences, which has the advantage of being easily maintained, updated and re-used, (iii) we report several experiments with a detailed evaluation of the proposed methods.

This paper is organized as follows: Sect. 2 discusses evidence-based medical guidelines and several existing methods for the identification of evidence classes,

and explores their limitations. Section 3 presents our approach to extend the existing algorithms to cover the identification of all evidence classes and formalizing the knowledge in a rule-based approach. Section 4 reports our experiments with the evaluation. Section 5 discusses future work and draws the conclusions.

2 Evidence-Based Medical Guidelines and Identification of Evidence Quality

Evidence-based medical guidelines are based on published scientific research findings. Those findings are usually found in medical publications such as those in PubMed. The articles selected are evaluated by an expert for their research quality, and graded in using the following classification system in five classes [8,9]: Class A1: Systematic reviews (i.e. research on the effects of diagnostics on clinical outcomes in a prospectively monitored, well-defined patient group); Class A2: High-quality randomised comparative clinical trials (randomised, double-blind controlled trials) of sufficient size and consistency; Class B: Randomised clinical trials of moderate quality or insufficient size, or other comparative trials (non-randomised, comparative cohort study, patient control study); Class C: Non-comparative trials, and Class D: Opinions of experts, such as project group members.

Based on the classification of evidences, we can classify the conclusions in the guidelines, alternatively called *guideline items*, with an evidence level. The following evidence levels on guideline items, are proposed in [8]: Level 1: Based on 1 systematic review (A1) or at least 2 independent A2 reviews; Level 2: Based on at least 2 independent B reviews; Level 3: Based on 1 level A2 of B research, or any level C research, and Level 4: Opinions of experts.

Here is an example of the conclusion in an evidence-based clinical guidelines in [8]:

```
Level:     1
Statement: The diagnostic reliability of ultrasound with an uncomplicated cyst
      is very high.
Evidence:  A1 Kerlikowske 2003. B Boerner 1999, Thurfjell 2002, Vargas 2004
```

which consists of an evidence level, a guideline statement, and their evidences with evidence classes.

In [12] Rosenfeld and Shiffman propose a notion of *strong recommendation* which is based on the evidence support by randomized controlled trials (RCT), which are shown in Fig. 1. Namely, strong recommendations are those which are supported by A-class evidences or some B-class evidences.

Rosenfeld and Shiffman suggest the following strategies or search filters of known validity such as the Cochrane highly sensitive strategy for randomized trials in PubMed that searches for:

1. "randomized controlled trial" or "controlled clinical trial" as publication type, or
2. "randomized," "placebo," or "randomly" in the abstract, or

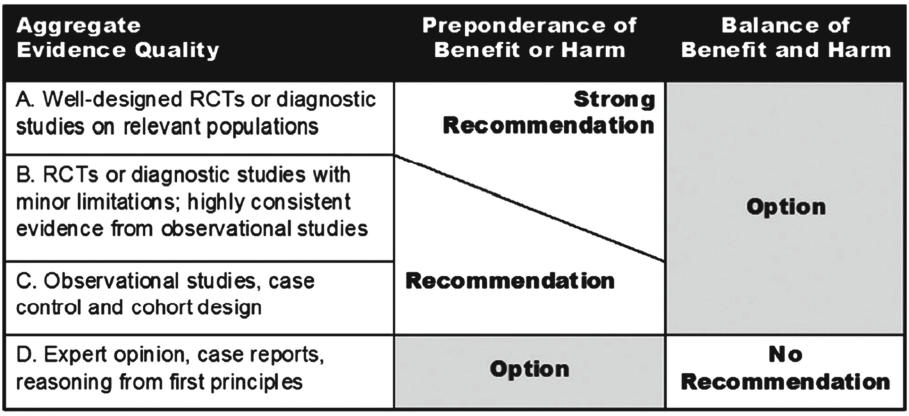


Fig. 1. Strong recommendation based on RCT evidences

3. "clinical trials as topic" as a Medical Subject Heading (MeSH) term, or
4. "trial" in the title, and restricts the final set (1 or 2 or 3 or 4) by excluding "animals" as MeSH term.

Reinders et al. [10] present a revised Rosefeld and Shiffman Algorithm to check whether or not a paper in PubMed is a strong evidence by Algorithm 1.1.

In [1] Cohen et al. describe a system that suggests updates to systematic reviews. Systematic reviews are similar to guidelines, in the sense that they attempts to summarize the scientific field surrounding a certain treatment. The difference is that while guidelines aim to give medical practitioners recommendations for a certain type of treatment, systematic reviews stick to summarizing the evidence in the domain. In [7], the Hedges team develops an optimal search strategy in Medline for retrieving systematic reviews, namely A1-class evidences. The Hedges Algorithm checks if "systematic review" is marked in its meta data. The checked terms also include a number of variants for the review, including "meta-analysis" (whether or not from a systematic review), and "review, academic"; "review, tutorial"; "review literature"; as well as separate terms for articles that often include reviews, such as "consensus development conference", "guideline", and "practice guideline".

The Hedge Algorithm (with some modification) is shown in Algorithm 1.2

We have conducted several experiments with Dutch Breast Cancer Guidelines (2004 and 2012) as test data for identifying the evidence quality. The advantage of using that test data is that the evidence classes of the evidences in those two guidelines are known. Thus, it can serve as a gold standard to evaluate the proposed algorithms, although there exist some errors in those assigned evidence classes.

We have applied Rosenfeld and Shiffman Algorithm to those evidences in the two Dutch Breast Cancer Guidelines to check whether or not the Algorithm can identify the evidence classes. Namely, for a give evidence in the guidelines,

Algorithm 1.1. Revised Rosenfeld & Shiffman Algorithm

Input: article a with metadata
Output: strong, boolean that indicates whether or not the article fulls Rosenfeld and Shiffman criteria

---

if "randomized controlled trial" ∈ a.pubtypes **then**
  c1= True
else if "controlled clinical trial" ∈ a.pubtypes **then**
  c1=True
end if
if "randomized" or "randomly" or "placebo" ∈ a.abstract **then**
  c2=True
end if
if "trial as topic" ∈ a.MeSHterms **then**
  c3=True
end if
if "trial" ∈ a.title **then**
  c4=True
end if
if "animals" ∈ a.MeSHterms **then**
  c5=True
end if
strong = (not c5 and (c1 or c2 or c3 or c4))
return strong

---

we obtain the PMID of the evidences. We use the PMID to get the meta-data (i.e., Publication Types, Abstract, Title, and MeSH Terms, and others) of the PMID from PubMed. The proposed algorithm checks the meta-data and estimates the evidence class of PMID. Table 1 is the result of Rosenfeld and Shiffman Algorithm, which shows how many of the originally known evidence-class of evidences have been estimated into strong evidence. The first two columns are the number of evidences in our golden standard with the corresponding class of evidence. For example, the second row in the table shows how many A1-class evidences have been estimated to be a strong evidence. The table shows that Rosenfeld and Shiffman Algorithm can estimate the strong evidences with recall= 0.4351 and precision=0.8189 if the strong evidences are interpreted as A1-class evidences, A2-class evidences, and B-class evidences. There are 104 (40+47+17) correct estimated strong evidence from the 239 (66+76+97) of the golden standard, which results in a recall of 0.435. In particular, Rosenfeld and Shiffman Algorithm can detect A-class evidences with recall=0.6061 for A1-class and recall=0.6184 for A2-class. However, Rosenfeld and Shiffman Algorithm is poor to estimate B-class evidences with recall=0.18.

We have also applied Hedges Algorithm to those evidences in the two Dutch Breast Cancer Guidelines. Table 2 is the result of Hedges Algorithm. That shows that Hedges Algorithm can detect the systematic review (with recall= 0.8485 and precision=0.37) if the systematic reviews are interpreted as the A1-class only. However, 61.84 percent of A2-class evidences are claimed to be a systematic

Algorithm 1.2. Revised Hedges Algorithm

Input: article a with metadata
Output: systematic review, boolean that indicates whether or not the article fulls Hedges criteria for the systematic review

---

if "randomized controlled trial" or"controlled clinical trial" or"systematic review" or"meta analysis" ∈ a.pubtypes **then**
  c1= True
end if
if "randomized" or"placebo" or "systematic review" or"meta analysis" ∈ a.abstract **then**
  c2=True
end if
if "randomized" or"placebo" or"systematic review" or"meta analysis" ∈ a.MeSHterms **then**
  c3=True
end if
if "randomized" or"placebo" or"systematic review" or "meta analysis" ∈ a.title **then**
  c4=True
end if
if "animals" ∈ a.MeSHterms **then**
  c5=True
end if
systematicreview = (not c5 and (c1 or c2 or c3 or c4))
return systematicreview

Table 1. Strong evidences estimated by Rosenfeld and Shiffman Algorithm.

Golden standard			
Class	Evidence number	Estimated strong evidence number	Recall
A1	66	40	60.61 %
A2	76	47	61.84 %
B	97	17	18 %
C	118	23	19.49 %
D	1	0	0 %
		Recall (A1,A2,B)	Precision (A1,A2,B)
Total	358	0.4351	0.8189

review because the algorithm consider the terms about randomized controlled trials as the indicators for the systematic review without the consideration of other quality features.

From the experiments above, we know that Hedges Algorithm can estimate the systematic review very well for A1-class, however, with the low precision for the A2-class evidences. Furthermore, Rosenfeld and Shiffman Algorithm can estimate the strong evidences reasonably well (high precision, but low recall).

Table 2. Systematic reviews estimated by Hedges Algorithm

Golden standard			
Class	Evidence number	Estimated systematic review number	Recall
A1	66	56	84.85 %
A2	76	47	61.84 %
B	97	20	20.62 %
C	118	27	22.88 %
D	1	0	0 %
		Recall (A1)	Precision (A1)
Total	358	0.8485	0.37

However, those two approaches are still too rough to cover all of the evidence classes.

We expect any new approach to have the following features.

- Fine-grained. Simple Boolean answers to a systematic review and a strong evidence are too rough to identify the evidence quality. We need an algorithm which can provide a more fine-grained distinction among the evidences, which can cover all of the evidence classes.
- Re-usability. We hope that the knowledge for identifying the evidence quality can be re-usable. It should not be necessary to change the source codes if we want to modify the existing knowledge or add some new knowledge for the evidence level identification.

3 An Approach to Identifying All Evidence Classes

In this section, we propose an approach to identifying all evidence classes. Our method of classifying the evidence is based on the A1, A2, B, C, D definitions as given in Sect. 2. We exploit the abstract, the title, and annotated MESH-terms of a paper for determining whether a paper can be classified as A1, A2, B, C or D.

For the A1 class we need to identify the systematic reviews. We take into account variations of the term 'systematic review' like 'meta analysis', 'meta-analysis', 'Systematic Review', 'Meta Analysis', 'Meta-analysis', 'Meta-Analysis as Topic'.

The A2-class we need high-quality randomised comparative clinical trials. In the current method we only identify randomised comparative clinical trials. We identify those by 'randomized controlled trial', 'controlled clinical trial', 'Randomized Controlled Trial', 'Controlled Clinical Trial', 'Randomized controlled trial', 'Controlled clinical trial', 'Randomized Controlled Trials as Topic', 'random allocation', 'double-blind method', 'single-blind method'. At this moment we neglect the "high-quality" class.

The B-class evidence is based on randomised clinical trials of moderate quality or insufficient size, or other comparative trials (non-randomised, comparative

cohort study, patient control). We identify randomised clinical trials by 'randomized', 'randomization', 'randomly', etc. In the current implementation we did not consider the quality and size of the trial. For determining a comparative trials we use again variations of terms, like 'comparative', 'controlled', 'placebo', 'Double-blind', 'compare', 'Placebo', 'Double blind', 'Comparative Study', 'Case-Control Studies', 'case control'. For cohort study we use 'cohort', 'cohort study', 'Cohort studies', 'Retrospective Studies', 'Prospective Studies'. For control study we use terms like 'control study', 'patient control study'.

The D-class (Opinions of experts) are found by those subjective statements with the phrases such as 'should be considered' and 'can be considered'.

Furthermore, for the re-usability, we propose a rule-based approach for the identification of evidence quality classification which enable us to adapt the definitions easily. The rule-based formalization of knowledge also makes it possible to accommodate natural language processing tools to represent and detect the varieties of terms (morphism). Moreover, the rule-based formalization also has distinguished advantages for easy maintenance, update, and re-usability. Through this rule-based approach it is not necessary to change the source codes if we want to modify the existing knowledge or add some new knowledge for the identification of quality levels of evidence [3]. We will show that it is easy to use this rule-based approach to accommodating the Hedges algorithm [1] and Rosenfeld and Shiman algorithm [12].

We consider the logic programming language Prolog as the rule-based language for the formalization. Prolog is a general purpose logic programming language associated with artificial intelligence, in particular, for knowledge representation and reasoning [16]. In Prolog, program logic is expressed in terms of relations. Those relations are formalized as a set of the predicates, like those in first order logic. A computation is initiated by running a query over these relations. In Prolog, the relations are represented as an atom which consists of a predicate with several terms as its parameters. A rule in Prolog has the following form

```
Head :- Body.
```

Where Head is an atomic formula, and Body is a conjunctive list of atomic formulas which are separated with commas and ends with a dot.

We introduce the predicate $evidenceClass(PMID, Class)$ to denote that the evidence class of the evidence (e.g., a PubMed Identifier PMID) is of a certain $Class$. According to the evidence classification described in the previous section, we propose the following simplified definition for the evidences:

```
evidenceClass(PMID,'A1'):-systemicReview(PMID).
%System reviews are A1-class evidences.

evidenceClass(PMID,'A2'):-highQualityTrial(PMID),randomizedComparativeTrial(PMID).
%Those based on high quality randomized trials are A2-class evidences

evidenceClass(PMID,'B'):-moderateQualityTrial(PMID),randomizedTrial(PMID).
%Randomized controlled trials of moderate quality are B-class evidences

evidenceClass(PMID,'B'):-insufficientSizeTrial(PMID),randomizedTrial(PMID).
```

```
%Randomized clinical trials with insufficient size are B-class evidences.

evidenceClass(PMID,'B'):-not(randomizedTrial(PMID)),comparativeTrial(PMID).
%Non-randomised comparative trials are B-class evidences

evidenceClass(PMID,'B'):- not(randomizedTrial(PMID)),comparativeCohortStudy(PMID).
%Non-randomised comparative cohort study are B-class evidences

evidenceClassPMID,'B'):-not(randomizedTrial(PMID)),controlStudy(PMID).
%Non-randomised patient control study are B-class evidences

evidenceClass(PMID,'C'):-not(comparativeTrial(PMID)),trial(PMID).
%Non-comparative trials are C-class evidences

evidenceClass(PMID,'C'):-not(comparativeTrial(PMID)),cohortStudy(PMID).
%Non-comparative cohort study are C-class evidences

evidenceClass(PMID,'D'):-opinionOfExpert(PMID).
%Those based on the opinion of experts are D-class evidences
```

Similar to Rosenfeld and Shiffman Algorithm and the Hedges Algorithm, we check the evidence quality by checking the terms that appear in the PMID metadata (i.e., the publication type, abstract of the article, MeSH terms, the article title, etc.) only. In the current work, we ignore the quality checking based on the patient cohort size and other quality indicators, and we will leave these for future work. Thus, we can define systematic reviews as those whose meta-data contain systematic review terms, like these:

```
systemicReview(PMID):-pubtype(PMID, Pubtypes),systemicReview(Pubtypes).
systemicReview(PMID):-abstract(PMID, Abstract),systemicReviewText(Abstract).
systemicReview(PMID):-meshTerms(PMID, MeSHTerms),systemicReview(MeSHTerms).
systemicReview(PMID):-title(PMID, Title),systemicReviewText(Title).
systemicReview(Pubtypes):-systemicReviewTerm(Term),member(Term,Pubtypes).
systemicReviewText(Abstract):-systemicReviewTerm(Term),substring(Term,Abstract).

systemicReviewTerm("systematic review").
systemicReviewTerm("meta analysis").
......
```

Furthermore, we check the randomized comparative trials for A2-class evidences by introducing the following rules:

```
randomizedComparativeTrial(PMID):-pubtypes(PMID,Pubtypes),
                randomizedComparativeTrialItem(Pubtypes).
randomizedComparativeTrial(PMID):-abstract(PMID,Abstract),
                randomizedComparativeTrialText(Abstract).
randomizedComparativeTrial(PMID):-meSHTerms(PMID, MeSHTerms),
                randomizedComparativeTrialItem(MeSHTerms).
randomizedComparativeTrial(PMID):-title(PMID, Title),
                randomizedComparativeTrialText(Title).
randomizedComparativeTrialItem(List):-randomizedComparativeTrialTerm(Term),
                member(Term, List).
randomizedComparativeTrialText(Text):-randomizedComparativeTrialTerm(Term),
               substring(Term, Text).
```

```
randomizedComparativeTrialTerm('randomized controlled trial').
randomizedComparativeTrial('controlled clinical trial').
......
```

However, the rules above are not sufficient to check the terms about "randomized" and about "controlled" separately. Thus, we also need the rules to check the conjunctive occurrence of the terms for randomized trials and the terms for comparative trials. These rules can also be used to check the B-class evidences. We can define the rules for C-class evidences and D-class evidences in a similar fashion. To make our results fully reproducible by others, the complete set of the rules (i.e. the Prolog program) can be downloaded from http://wasp.cs.vu.nl/sct/download/evidenceQuality3b.pl. For any PMID, the Prolog program will access the PubMed data online and return the corresponding evidence class.

4 Experiments and Evaluation

Our third experiment is to apply our more fine grained Algorithm to estimate the five quality classes (A1,A2,B,C,D) of the evidences. The results are shown in Table 3, where we take the five groups of evidences as five sets of test data. For the 66 original A1-class evidences, 16 evidences can be detected correctly. For 76 original A2-class evidences, 33 evidences can be detected correctly. For 97 original B-class evidences, 22 evidences can be detected correctly. Similarly, we can find the results for those original C-class evidences and D-class evidences. The system returns $(26 + 100 + 56 + 115 + 3) = 300$ certain answers if we do not count those unknown answers. Thus, the precision of the Algorithm is $(16+33+22+52+1)/300 = 124/300 = 0.4133$, and the recall of the Algorithm is $(16 + 33 + 22 + 52 + 1)/358 = 0.3464$. If we consider the 66 original A1-class evidences only, the Algorithm can detect 16 ones, giving recall=$16/66 = 0.24$, and precision=$16/(16+37+6+3) = 0.22$. We are more interested in the capability of the Algorithm for detecting the top-class (i.e. A1+A2) evidences. For the detection of the top-class evidences, the rule-based Algorithm has recall=$(19 + 70)/142 = 0.63$ and precision=$(19 + 70)/(19 + 70 + 15 + 17) = 0.74$.

It is not a surprise that the rule-based Algorithm can estimate only 24.24 percent of A1-class evidences, because our rules for A1-class evidences are defined only by checking the systematic review terms such as "systematic review" and "meta-analysis", without the consideration of its quality with respect to the patient cohort size and the analysis results. In the near future we take into account the cohort size and the quality of the results. We expect that this will improve the classification of A1, A2 and B classes. Furthermore, we do not want to loose the condition that covers the terms for randomized controlled trials, like that used in Hedges Algorithm, which may include many non-A1 evidences. Notice that if we consider only the class A1 and A2 results, then both recall and precision are quite good.

Our fourth experiment is to test the usability of the proposed approach. More exactly, we use the fine grained classification algorithm for the guideline update of Dutch Breast Cancer Guideline (2012). We have implemented a guideline

Table 3. Estimated evidence classes by Rule-based approach

Class	Evidence number	A1	A2	B	C	D	Unknown	Recall	Precision
A1	66	16	37	6	3	0	4	0.2424	0.2222
A2	76	3	33	9	14	0	7	0.4342	0.5593
B	97	4	14	22	45	0	12	0.23	0.2588
C	118	3	16	19	53	2	24	0.45	0.5699
D	1	0	0	0	0	1	0	1	1
A1+A2	142	19	70	15	17	0	11	0.63	0.7355
Total	358	26	100	56	115	3	47	0.3464	0.4133

update component in SemanticCT [4], a semantically-enabled system for clinical trials. The SemanticCT system is powered by the LarKC platform, a semantic platform for scalable semantic data processing and reasoning [2]. That guideline update interface in SemanticCT provides various selection mechanisms for guideline designers and other researchers to find relevant research evidences on guideline content. The interface provides the selection options to select the formalized evidence-based medical guidelines with different topics (e.g. subsection title of the guideline document). For each selected topic, the interface will show a list of the guideline items with their evidence levels, referred evidences and their evidence classes. The conclusions of both guidelines have been converted into RDF NTriple data, and loaded into SemanticCT.

For each guideline item, the system can create a query by a combination of the terms which are used in the statement of the guideline item, post the query to search for the relevant evidences in PubMed. We use the rule-based algorithm to estimate the evidence classes of the found relevant evidences by SemanticCT.

We have selected 15 guideline items in the Dutch Breast Cancer Guideline (2012) for a test of guideline updating in SemanticCT. Table 4 shows the results of the rule-based algorithms for the estimation of evidence classes over the newly found evidences for those selected guideline items. For each guideline item, the system can find a set of relevant evidences (with max = 299 evidences, min = 8 evidences, and average = 134.4 evidences). Of those found evidences, the Algorithm can estimate the evidence classes for a significant amount of evidences (with max = 97.67 %, min = 56.20 %, and average = 75.65 %). The results justify the usability of the proposed method. It is also interesting to see that the most frequently occurring evidences are B-class ones.

5 Discussion and Conclusions

Finding relevant evidences for updating clinical guidelines has been studied by several groups. The teams of Tsafnat and others have been working on automatic methods in searching for the literature and creating systematic review for the update of clinical guidelines [14,15]. They develop the algorithms for re-creating

Table 4. Estimated evidence classes for found evidences in guideline update. Known: Evidences with an estimated evidence class (e.g., the sum of A1+A2+B+C+D)

Id	Found evidences	A1	A2	B	C	D	Unknown	Known	Percentage
12_2_1	121	1	6	19	39	3	53	68	56.20 %
12_2_2	101	6	6	54	16	0	19	82	81.19 %
12_3_1	292	3	13	115	80	1	80	212	72.60 %
12_4_1	31	1	0	12	11	0	7	24	77.42 %
12_5_1	264	4	10	117	94	0	39	225	85.23 %
12_5_10	206	6	60	53	34	0	53	153	74.27 %
12_6_2	157	2	5	56	36	1	57	100	63.69 %
12_6_3	30	1	2	11	8	0	8	22	73.33 %
12_9_1	49	1	3	14	14	1	16	33	67.35 %
12_9_3	8	0	0	4	3	0	1	7	87.50 %
12_9_4	39	0	1	11	15	0	12	27	69.23 %
12_12_1	226	2	6	102	66	0	50	176	77.88 %
12_12_4	150	3	6	81	29	0	31	119	79.33 %
12_13_2	43	0	4	19	19	0	1	42	97.67 %
12_13_4	299	10	10	106	88	1	84	215	71.91 %
Total	2016	40	132	774	552	7	511	1505	
Average	134.40	2.67	8.80	51.60	36.80	0.47	34.07	100.33	75.65 %

the systematic search whenever the guideline needs updating. In this paper, we have developed a method which covers not only the identification of systematic reviews, namely, A1-class evidences, but also covers other evidence classes, which include A2, B, C, D class evidences. In [11], meta-analyses are used to create citation networks of randomized controlled trials (RCT) addressing the same clinical questions. A primary measure is proposed to use the proportion of networks where following citation links between RCTs identifies the complete set of RCTs, forming a single connected citation group. Interesting future work is to investigate the network relation among evidences.

What we have done in our formulation of identification knowledge is just the term checking of evidences classes. We see several possibilities to improve our method in both precision and recall. One direction in particular in improvement of recall is the use of a Natural Language Processing tool to deal with the morphology and the lexicology to cover more varieties of the identification terms. This can easily be integrated with our rule-based approach, because the logic programming language Prolog provides the facility to integrate with the Natural Language Processing library. In Prolog, a definite clause grammar (DCG) is usually used to express grammars for natural or formal languages. Therefore DCG rules can be used to provide an easy and basic processing for the varieties

of terms as we already did. Another interesting future work is to develop a set of rules to identify the quality by checking the patient cohort sizes and their analysis. Namely we can formalize the predicates such as 'high-quality-trial', 'moderate-quality-trial', 'insufficient-size-trials', and others, so that our method can identify the evidence quality better. We expect that this will result in an improvement in precision of our method.

In this paper, we have proposed and implemented a more fine-grained evidence level identification by using a rule-based approach. That provides a convenient way to use the approach of DCG rules for the extensions. As we have discussed before, the rule-based approach provides the possibility for ease of maintenance, update, and re-usability of the identification knowledge about evidences. This work is a step in the direction of a faster guideline update process by giving guideline developers support by identifying the strength of the evidence. In the near future we will work on several improvements of our method, and further evaluation in collaboration with experts.

Acknowledgements. This work is partially supported by the European Commission under the 7th framework programme EURECA Project (FP7-ICT-2011-7, Grant 288048).

References

1. Cohen, A., Ambert, K., McDonagh, M.: Studying the potential impact of automated document classification on scheduling a systematic review update. BMC Med. Inf. Decis. Making **12**(1), 33 (2012)
2. Fensel, D., et al.: Towards LarKC: a platform for web-scale reasoning. In: Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2008). IEEE Computer Society Press, CA, USA (2008)
3. Huang, Z., ten Teije, A., van Harmelen, F.: Rule-based formalization of eligibility criteria for clinical trials. In: Peek, N., Marín Morales, R., Peleg, M. (eds.) AIME 2013. LNCS, vol. 7885, pp. 38–47. Springer, Heidelberg (2013)
4. Huang, Z., ten Teije, A., van Harmelen, F.: SemanticCT: a semantically enabled clinical trial system. In: Lenz, R., Miksch, S., Peleg, M., Reichert, M., Riaño, D., ten Teije, A. (eds.) ProHealth 2012 and KR4HC 2012. LNCS, vol. 7738. Springer, Heidelberg (2013)
5. Iruetaguena, A., et al.: Automatic retrieval of current evidence to support update of bibliography in clinical guidelines. Expert Syst. Appl. **40**, 2081–2091 (2013)
6. Becker, M.E.M., Neugebauer, E.A.M.: Partial updating of clinical practice guidelines often makes more sense than full updating: a systematic review on methods and the development of an updating procedure. Expert Syst. Appl. **67**, 33–45 (2014)
7. Montori, V.M., Wilczynski, N.L., Morgan, D., Haynes, R.B., Team, H.: Optimal search strategies for retrieving systematic reviews from medline: analytical survey. BMJ **330**(7482), 68 (2005)
8. NABON. Breast cancer, dutch guideline, version 2.0. Technical report, Integraal kankercentrum Netherland, Nationaal Borstkanker Overleg Nederland (2012)
9. NSRS. Guideline complex regional pain syndrome type I. Technical report, Netherlands Society of Rehabilitation Specialists (2006)

10. Reinders, R., ten Teije, A., Huang, Z.: Finding evidence for updates in medical guideline. In: Proceedings of the 8th International Conference on Health Informatics (HEALTHINF2015), Lisbon, 11–15 January 2015
11. Robinson, K., Dunn, A., Tsafnat, G., Glasziou, P.: Citation networks of trials: feasibility of iterative bidirectional citation searching. J. Clin. Epidemiol. **67**(7), 793–799 (2014)
12. Rosenfeld, R., Shiffman, R.: Clinical practice guideline development manual: a quality-driven approach for translating evidence into action. J. Am. Acad. Otolaryngol.-Head Neck Surg. **140**(6 Suppl 1), S1–S43 (2009)
13. Shekelle, P.G., Woolf, S.H., Eccles, M., Grimshaw, J.: Developing guidelines. BMJ: Br. Med. J. **318**(7182), 593–596 (1999)
14. Tsafnat, G., Dunn, A., Glasziou, P., Coiera, E.: The automation of systematic reviews. BMJ: Br. Med. J. (2013)
15. Tsafnat, G., Glasziou, P., Choong, M., Dunn, A., Galgani, F., Coiera, E.: Systematic review automation technologies. Syst. Rev. **3**, 74 (2014)
16. Wielemaker, J., Huang, Z., van der Meij, L.: SWI-prolog and the web. J. Theory Prac. Logic Program. **8**, 30 (2008)

Answer Set Programming for Temporal Conformance Analysis of Clinical Guidelines Execution

Matteo Spiotta[1,2], Paolo Terenziani[1], and Daniele Theseider Dupré[1(✉)]

[1] DISIT, Sezione di Informatica, Università del Piemonte Orientale, Vercelli, Italy
dtd@di.unipmn.it
[2] Dipartimento di Informatica, Università di Torino, Torino, Italy

Abstract. Analyzing conformance of execution traces with clinical guidelines is not a trivial task, because guidelines are developed for physicians who should always interpret them according to their general knowledge; their application to the specific conditions of a patient or a specific context cannot always be foreseen in the guideline specification. In this paper we consider the conformance problem not only for the sequence of action execution events, but also for their timing: the guideline may include temporal constraints for the execution of actions, and its adaptation to a specific patient and context may add or modify conditions and temporal constraints for actions. We propose an approach for analyzing execution traces in Answer Set Programming with respect to a guideline and Basic Medical Knowledge, pointing out discrepancies – including temporal discrepancies – with respect to the different knowledge sources.

1 Introduction

A Clinical Guideline (CG) is "a systematically developed statement to assist practitioner and patient decisions about appropriate health care for specific clinical circumstances" [7]. CGs are developed in order to capture medical evidence and to put it into practice, and deal with "typical" classes of patients, since the CG developers cannot define all possible executions of a CG on any possible specific patient in any clinical condition. When treating "atypical" patients, physicians have to resort to their Basic Medical Knowledge (henceforth BMK; informally, in this paper, BMK includes the different forms of medical knowledge that physicians have acquired during their studies and their clinical practice).

The interplay between CG and BMK recommendations can be very complex. For instance, actions recommended by a CG could be prohibited by the BMK, or a CG could force some actions despite the BMK discourages them (see, e.g., [3]). Such a complexity significantly increases in case the temporal dimension is taken into account: indeed, (i) temporal information is an intrinsic part of most CGs and BMK, and (ii) the interplay between CGs and BMK occurs in time. Regarding issue (i), actions have pre-conditions which temporally constrain them

This research is partially supported by Compagnia di San Paolo.

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 65–79, 2015.
DOI: 10.1007/978-3-319-26585-8_5

(e.g., an action must be performed only while a precondition holds), and may be temporally constrained with each other (e.g., in case of hip fracture, surgery is recommended within 36 h after admission). Considering (ii), though there are cases in which CG and BMK recommendations are in contrast, and one of the two prevails over the other, which is then ignored, in most cases the two recommendations should be "merged" along time. Typical cases are the treatment of exceptions [9,12,13], which, depending on the situation, may be treated as soon as they occur (thus suspending CG execution), delayed after the end of the CG, or executed concurrently with it. In all cases, some of the temporal constraints (in the CG and/or in the BMK) may be violated.

Unfortunately, the proper solution for managing the interplay between CGs and BMK is usually situation- and patient-dependent, and, in general, expecting that a model could provide such solutions is not realistic. Nevertheless, computer science can support physicians in the analysis of such an interplay, considering also patient data and contextual information, providing physicians with the relevant knowledge to understand and evaluate how the patient has to be treated. This is the challenging goal of the approach in this paper. In particular, as in [14], we explore the interplay between CGs and BMK from the viewpoint of *a posteriori* conformance analysis [10], intended as the adherence of an observed CG execution trace to both the CG and BMK. We do not provide any form of evaluation of how the interplay has been managed (i.e., whether the treatment was appropriate or not): we aim at identifying, in the trace, situations in which some recommendation (either in the CG or in the BMK) has not been followed, and at providing possible justifications for situations of non-conformance. In such a way, conformant treatments (to both the CG and the BMK) are automatically identified and presented to physicians, as well as non conformant treatments and their possible justifications, based on the available knowledge.

2 Input to the Framework

At least four different types of data/knowledge sources should be considered to analyze compliance: patient data, contextual data, CG model, and BMK.

We distinguish two types of **patient data**. We consider *patient findings*, i.e., data which are usually collected in patients' EHR. In particular, we assume that all data required during patient treatment are available, and that all such pieces of information are temporally tagged. We also assume to have a complete *log* of all the clinical actions executed on the patient, in which each occurrence of actions is temporally tagged. Specifically, we assume that the starting and ending time (both in case of completion and of abort) of all actions are recorded.

We consider **contextual data** such as personnel and resources availability.

Our approach is not biased towards any specific **CG formalism**; however, we will use the GLARE formalism [1,17] as a concrete example, due to its specific attention to the temporal aspects. Indeed, we consider the possibility of distinguishing between atomic and composite actions, and of specifying (therapeutic and diagnostic) decisions. CGs specify the control flow of actions and include

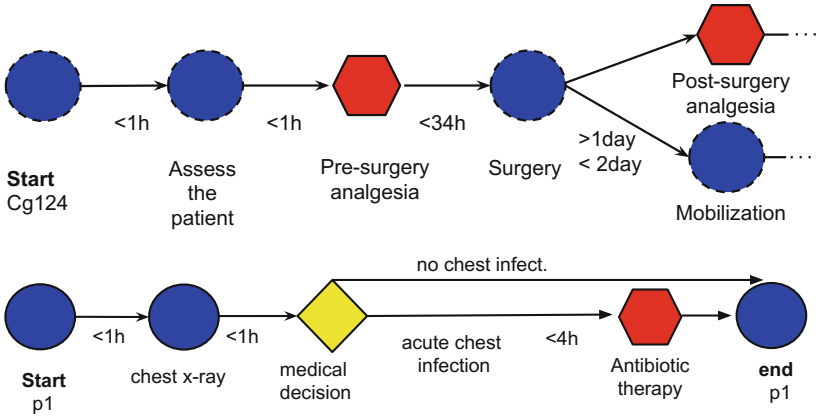


Fig. 1. Hip fracture CG (above) and chest infection plan (below). Circles are atomic actions, hexagonal nodes are composite actions, diamond nodes are decisions.

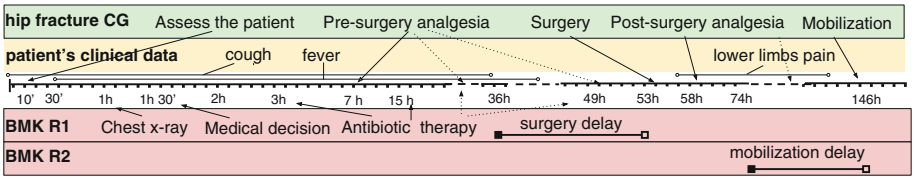


Fig. 2. Example case.

temporal constraints between them. Additionally, actions may have preconditions, and temporal constraints between the time when preconditions hold and the time when the related action must be executed can be specified. Specifically, in this paper (as in GLARE) we assume that temporal constraints may be used to impose a minimum and maximum delay between the starting/ending points of events (actions and/or preconditions in our context).

Figure 1 presents an adapted excerpt, represented in GLARE, of the guideline for hip fracture by the British National Institute for Health and Care Excellence (NICE) [6]. The CG contains information about recommended timing to ensure the effectiveness of the treatment. Since it recommends that surgery is performed "on the day of, or the day after, admission" we consider a "within 36 h" recommendation. Rehabilitation through mobilization therapy must start the day after surgery.

We assume that pieces of knowledge in the **BMK** are formed by:

- a *trigger,* i.e., conditions on the patient and context that make the piece of knowledge relevant, and either:
- a simple or composite action, which is suggested in the given conditions; or:
- knowledge (such as counterindications) suggesting to avoid or delay some action, in one of the following forms:

- *Avoid a:* states that action a should not be executed (we assume that such a statement is triggered by conditions that are not reversible);
- *Delay a while c:* states that action a should not be performed as long as c holds (where we expect c to be reversible);
- *Delay a for d:* suggests delaying action a for time d.

Knowledge involving suggested actions (**"do"** knowledge for short) is similar to exceptions in [9] and guideline-independent exceptions in [12], whose triggers are not expected to be only relevant at a specific point of a CG, but rather may occur – requiring treatment – at any point. Both "do" knowledge and the one that suggests avoiding or delaying an action (**"do not"** knowledge for short) refine rules proposed in [14], taking better into account the temporal dimension.

Then, while the CG provides treatment for general cases, the BMK may account both for additional actions and for cancellation or a different timing of actions prescribed by the CG (see Sect. 3.1).

The clinical case used to present our approach is the treatment of a patient which has been hospitalized to treat a hip fracture. At hospitalization time, the patient had also cough and high temperature. The following BMK rules R1 and R2 relevant to the case are considered:

R1. For patients with high body temperature and cough, the presence of a chest infection has to be investigated through a chest x-ray, and, if present, treated with an antibiotic therapy[1] (see Fig. 1).
R2. Mobilization has to be delayed for patients having pain in lower limbs.

Figure 2 describes an example case. Actions in the first row are directly recommended by the CG. However, several non-compliant actions (with respect to the CG) appear. Actions *chest x-ray*, *medical decision*, *antibiotic therapy* are justified by rule R1. The fact that another problem is being treated should also explain the delay of *hip surgery* beyond the 36 h recommended by the CG, even though we do not expect to have an explicit model of the condition which allows surgery to proceed. Rule R2 is triggered because the day after surgery the patient has pain in the lower limbs; it justifies the delay of *mobilization*.

3 Execution Model

In order to establish whether a trace is adherent to the guideline, the semantics of the guideline should be established. Possible states and transitions of a work action are as in Fig. 3. The control flow of the CG execution, or triggers in the BMK, may indicate that a given action has to be considered for execution (is a candidate). A candidate action could become active, i.e., actually started, or discarded

[1] Actually, the recommendations in [6] explicitly mention acute chest infection as one of the conditions to be checked and, if necessary, treated, to avoid delaying surgery too much; but they also mention, without detailing them, less common concerns which may require delaying surgery: we consider chest infection like one of these.

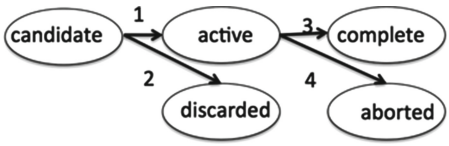


Fig. 3. States for a work action

(transitions 1 or 2); if active, it could either be completed or aborted (transitions 3 or 4). In order to define which transition could occur and when, we start discussing the case of a single source of knowledge, the CG, and, for the sake of brevity, we limit the discussion to transitions 1 and 2.

The control flow that makes a candidate imposes constraints on the time t_{act} when action a could become active. In general, such constraints are given with respect to the start or end times of previous actions in the control flow (possibly more than one action). In order to simplify the description in the following, we assume they are given with respect to the time t_{ca} when the action becomes candidate and are of the form:

(i) $t_{ca} + m \leq t_{act} \leq t_{ca} + n$

Then, $t_{ca} + n$ provides a deadline for starting the action in order to conform to the CG. In addition, preconditions on a impose constraints of similar form:

(ii) $t + m \leq t_{act} \leq t' + n$

where t and t' are the start or end time of a precondition, or, more precisely, of an episode of a precondition. Expressions $t' + n$ in constraints (ii) do not, in general, provide deadlines because the precondition could become true again[2]; e.g., a blood exam could be constrained to be executed within 1 day after the previous step (type (i)) and could require fasting for at least 8 h (type (ii)).

The conformant execution after an action becomes candidate can be characterized as follows:

1. The action should start (become active) at a time t_{act} such that all preconditions, with their temporal constraints, enable the action, if one such time exists.
2. Otherwise, when the first deadline is reached, the action is discarded.

The conditions above can be represented in first-order logic, using the following predicates:

– $\mathit{preconditions}(a, \overline{t})$ means that an episode for each precondition of a hold, and $\overline{t}$ are the start and end times of such episodes;

[2] In some cases there is a deadline, e.g. for a diagnostic action that should be performed within a given time after the *first* episode of a symptom. Such deadlines can be dealt with as the other ones, but, again, for simplicity, we ignore them in the following.

- $C(t_{ca}, t_{act}, \bar{t})$ represents the constraints between the time t_{ca} when the action becomes candidate, the times $\bar{t}$ of start and end of precondition episodes, and the time t_{act} when the action could become active;
- *trans* represents times of transitions and their binding for the same action instance; in particular, $trans(a, candidate, t_{ca})$ means that a becomes candidate at t_{ca}, while $trans(a, t_{ca}, active, t_{act})$ (with four arguments) means that the instance of a that became candidate at t_{ca} becomes active at t_{act}.

The correspondents of conditions (1–2) above are then:

$$\forall t_{act} t_{ca} trans(a, t_{ca}, active, t_{act}) \rightarrow \exists \bar{t}\ preconditions(a, \bar{t}) \wedge trans(a, candidate, t_{ca}) \wedge C(t_{ca}, t_{act}, \bar{t}) \quad (1a)$$

$$\forall t_{ca} trans(a, candidate, t_{ca}) \wedge [\exists t_{act} \bar{t}\ preconditions(a, \bar{t}) \wedge C(t_{ca}, t_{act}, \bar{t})] \rightarrow \exists! t_{act}\ trans(a, t_{ca}, active, t_{act}) \quad (1b)$$

$$\forall t_{ca}\ trans(a, candidate, t_{ca}) \wedge [\nexists t_{act} \bar{t}\ preconditions(a, \bar{t}) \wedge C(t_{ca}, t_{act}, \bar{t})] \leftrightarrow trans(a, t_{ca}, discarded, t_{ca} + n) \quad (2)$$

Formula (1a) states that if the action starts at t_{act}, it is allowed to start at such time; (1b) states that if there are times when a candidate action is allowed to start, it starts (at one of those times, due to (1a)). Formula (2) states that the action is discarded (at time $t_{ca} + n$, i.e., at the deadline) if and only if there is no allowed time to start it.

Such formulas, and similar ones for the other transitions, can be seen as a way of providing semantics for the CG formalism, and are the basis for detecting (non-)compliance of a sequence of events with the guideline; in fact, a non-compliance corresponds to the fact when one of such formulas is false, and types of non-compliance correspond to different ways such formulas can be false.

Item 2 and Formula (2) correspond to a strict interpretation of the CG recommendations. If not all conditions for starting the action can be met, there may be valid medical alternatives to discarding the action: either relaxing a precondition or a deadline, i.e., performing the action late, or even performing the action at a time when a precondition recommends not to. We therefore point out the occurrence of case 2, and for a non-conformance case where a deadline or a precondition is violated, it could be pointed out whether performing the action meeting *all* the recommendations was possible or not.

3.1 Taking BMK into Account

In the following we describe the alterations of the guideline execution that can possibly be considered justified, taking BMK into account. We identify several situations where CG and BMK recommendations could be merged, or, in case they are in contrast, one could override the other.

When a (possibly composite) action a in the *"do"* BMK knowledge is triggered, several **modalities** of execution are considered possible (similarly to [12]):

- execution of a and the CG proceeds concurrently, according to their own constraints (**concur** modality);
- a and the CG are executed concurrently, but temporal constraints are not enforced (since they are presumably given for the case where there is no concurrent treatment for another problem) (**concur_no_tc**); as special cases, a is delayed after the end of the CG execution (**after**), or a is executed first, and then the CG execution proceeds (**before**);
- the CG execution continues, and the BMK suggestion is ignored (**ignore**);
- the execution of the CG is aborted and a is executed (**abort**).

A special case occurs for the concurrent modality, in case the same action b is candidate for both CG execution and the execution of the BMK action: temporal constraints from both the CG and BMK (**cg_bmk_constr**), or from either of them (**cg_constr, bmk_constr**), may be enforced.

When an action a is executable or active and a *"do not"* BMK rule is triggered, the options are as follows, depending on what is triggered:

- *Avoid a:* the action a is discarded, if candidate, or aborted, if active (**avoid**); or the BMK rule is ignored (**ignore_avoid**);
- *Delay a while c:* either the BMK rule is ignored (**ignore_delay**), or c is used as an additional precondition for action a (**add_delay**), or it replaces preconditions for a (**delay**).
- *Delay a for d:* either the BMK rule is ignored (**ignore_delay**), or it adds the constraint $t_{now} + d \leq t_{act}$ (**add_delay**), or replaces with it the constraints on t_{act}(**delay**).

This accounts for a wide range of modifications to the set of executions allowed by a CG, taking into account knowledge, the BMK, that may be *general*, not being related to the class of problems addressed by the CG, as well as *specific*, justifying adaptation of the CG to a specific class of patients, which is not explicitly considered in the CG definition. It allows one of the knowledge sources to prevail over the other one (which is then ignored), or for its recommendations to be given temporal priority on the other ones. The result can only approximate the set of medically correct adaptations of a CG to a case; it is a way of making conformance analysis more flexible, without assuming exhaustivity of CGs.

4 Conformance Analysis

The goal of conformance analysis in the context described in the previous section is to reconstruct whether the sequence of events in the log can be interpreted as an execution of the CG, with the possible alterations that take BMK into account. If an exact reconstruction is not possible, discrepancies with the log must be pointed out. However, in case a BMK rule is triggered, the alternatives in Sect. 3.1 introduce several potential reconstructions, also because some the modalities are (logically) stronger than others. For example, in a "do" rule, the *concur*, *after* and *before* modalities are strictly stronger than *concur_no_tc*; then,

if one of the former is consistent with the log, also *concur_no_tc* is. Similarly, *add_delay* is stronger than *delay* and *ignore_delay*. With different data, a weak modality may be consistent, while a stronger modality is not, i.e., it implies some discrepancies. In general, especially in case several BMK rules are triggered, this gives rise to several potential solutions, each one with zero or more discrepancies.

We introduce therefore a notion of preference among explanations, where, as a primary criterion, we prefer reconstructions with a minimum number of discrepancies with the log. Secondarily, i.e., among explanations with the same number of discrepancies, we prefer explanations that are conformant with more knowledge. In the first case mentioned above, we prefer *concur* over *after* and *before* (even though it is not logically stronger) since *concur* means executing the CG and the BMK plan respecting all their constraints; and we prefer *after* and *before* over *concur_no_tc*, since the former impose to respect the internal quantitative constraints of each plan, while the latter only imposes that the control flow of each plan is followed. The *add_delay* modality is preferred over *delay* and *ignore_delay*, since it corresponds to being conformant with more knowledge.

These preferences are not intended as medically preferred choices, but rather as a way for not presenting explanations that unnecessarily assume a deviation from prescriptions of the overall knowledge. The modality is part of the solution; this is useful especially in cases where being conformant with all knowledge is not possible, i.e., two pieces of knowledge provide contrasting prescriptions, and it is not specified which one should prevail. Consider the case where, according to a BMK rule, an action was actually delayed, overriding the CG constraints: in the interpretation of the log there is evidence of this choice, even though it was not necessarily the best medical choice. On the other hand, for a non-conformant action, in case a conformant execution was also possible, this is pointed out.

In order to perform such conformance analysis, formulae in Sect. 3 (i.e. including (1a, 1b and 2)) could be the basis for analyzing conformance of a trace with respect to a CG only. In fact, it should be detected whether some of the formulae are false for some action. E.g. (1b) is violated if a should be started, but it is not. In the negation of formula (1a) we could distinguish several cases: the action is started, but either it was not candidate, or some of its precondition is never true, or for the times when the action changes state, there are no episodes of the preconditions such that the constraints hold.

The set of formulae could be further elaborated in order to interpret traces based on the BMK as well as the CG: the actual preconditions and temporal constraints to enforce depend on the knowledge source(s) being considered, and, e.g., in (1a), the executed action may fail to be a CG candidate, but be a BMK candidate. We do not, however, describe in detail this option; rather, we describe in the following how the approach is represented in Answer Set Programming (ASP) [8], which is also based on logic; similarly to [14], this allows an ASP solver to infer whether and how the sequence of events in the log can be reconstructed according to the patient DB, the CG and the BMK, following the specifications in Sect. 3. While a SAT solver finds models of a propositional representation, an ASP solver (in particular, we used Clingo [18]) finds stable models of an ASP representation,

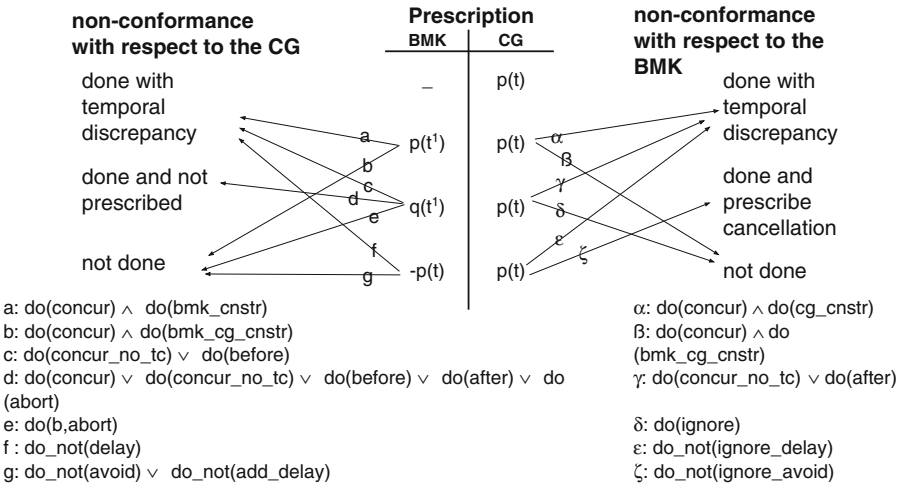


Fig. 4. Discrepancies and their justifications.eps

which is nonmonotonic and allows rules with variables, which are substituted in a grounding phase with a finite number of constants (in our case, for example, variables for time instants take values in a finite domain of time points). Annotation rules are used to identify cases of non-conformance with at least one source of knowledge, which include cases where formulae in Sect. 3 are false. The nonmonotonic nature of ASP is useful to model exceptions to guideline execution according to the BMK. Optimization statements in ASP are used to represent preferences.

Figure 4 summarizes an analysis which contributes to demonstrating coverage of our approach. It represents cases where a discrepancy with one knowledge source may be justified by the other source. Different lines in the table provide different cases as regards current prescriptions from the CG and the BMK ($p(t)$ means that action p is candidate to start at time t, while $-p(t)$ means that p should not be done at t). Arrows connect a line to a type of discrepancy with respect to the CG or the BMK, and labels below provide a shorthand description of what will be part of the answer set in that case. For example, if the CG and BMK propose the same action as candidate (2nd line), it may be the case (a) that the action is performed without conforming to the CG constraints (the exception runs in concurrent modality and only the bmk constraints are enforced), or (α) it is performed without conforming to the BMK constraints, or (b and β) it is not performed because all constraints are enforced, but they are never all true.

4.1 ASP Representation

The ASP model can be divided in the following main components:

- the **inputs** for the analysis: the log trace, the CG and BMK model;
- the **control flow component** which provides the set of allowed actions in each state (considering both the CG and the BMK);

- the **interaction component** which generates the different modalities of execution for the fired BMK rules (i.e., "concur","before", etc.);
- the **temporal constraints component** which determines the allowed timing of actions with respect to other actions and action preconditions;
- the **annotation rules component** which detects non compliance, and selects BMK execution scenarios which minimize unexpected behaviour.

The ASP model reconstructs for each state the behaviours allowed by the execution model. The sequence of actions in the log, together with patient and context data, makes it possible to identify the paths in the control flows (of the CG and the BMK plans) followed during execution. When a triggering condition is satisfied by context and patient data, a candidate interpretation (an answer set) is generated for each possible modality of rule execution. Preconditions and temporal constraints are evaluated taking into account the actions specification, with variations imposed by the BMK rules in the specific execution context. The annotation rules use this information to detect behaviours that differ from the expected one in each candidate interpretation.

The encoding of **inputs** is straightforward. Patient and context data are represented with ground facts (`holds(Name,Value,Ts,Te)`) binding data to intervals in which they were known to be true. The GLARE representation of CG and BMK plans is encoded with ground facts which describe the control flow, preconditions and time constraints. BMK trigger rules can be mapped to a set of ASP rules having the trigger condition as body and one of the following as head:

- `prescribe(ID,A,T)` for a BMK rule ID prescribing the composite action A at time T;
- `prescribeAvoid(ID,A,T)` for a BMK rule ID prescribing the discard/abort of an atomic action A at time T;
- `prescribeDelayWhileC(ID,C,A,T)` for a BMK rule ID prescribing at time T the delay of A until condition C becomes true.
- `prescribeDelay(ID,C,D,A,T)` for a BMK rule ID prescribing at time T the delay of A for time D given that condition C is true.

For example, the ASP encoding of R2 is:

```
prescribeDelayWhileC(r2,lower_limb_pain,mobilization,T):-
      holds(lower_limb_pain,Ts,Te),T>Ts,T<Te,candidate(mobilization,T).
```

The **control flow component** evaluates the control flow of CG and BMK plans. Similarly to [14], given the executed action and the control flow description, instances of `candidate(Src,A,Tca,T)` are inferred, which state that knowledge source `Src` (either `cg`, or the BMK rule identifier) enables the execution of `A` in `T` (the time `Tca` when it became candidate identifies an action instance).

The **temporal constraints component** reconstructs the expected timing of the actions taking in account the possible variations introduced by the BMK. A predicate `tc(Src,Tca,A,started/completed,Ts,Te)` specifies the allowed interval `[Ts,Te]` for starting or completing action `A` prescribed by knowledge source `Src` at `Tca`. In general we deal with STP constraints for actions in a group, but,

for the sake of brevity, we only show the case of temporal constraints given for the starting point wrt the end of previous action in the control flow:

```
tc(Src,Tca,A,started,Tca+Min,Tca+Max):-
   wf_tc(Src,A,started,B,completed,Min,Max),
   candidate(Src,A,Tca,_),trans(B,completed,Tca),
   not exception(Src,A,Tca,deleteWFtc;changeWFtc).
```

where `wf_tc` states that `A` must be started between `Min` and `Max` instants after `B` was completed; `candidate` and `trans` detect the action `B` that enabled `A` at time `Tca`; `exception`, defined in the BMK component, could prescribe variations to the CG time constraints, or cancel the CG constraints. If no temporal variations are required, the action should be performed in the interval `Tca+Min,Tca+Max`; otherwise the BMK component (below) either cancels the constraints (`deleteWFtc`), or it suggest a different timing (`changeWFtc`). In this last case `tc` is defined in the BMK component and the interval derived from the CG specification is used to check whether being conformant with both knowledge sources is possible. For the interval determined by the constraints in force, action preconditions are checked to detect whether the action should be started or discarded.

The **BMK component** determines the execution modality of BMK rules and their consequences. Some ASP rules are shown below for the "delay while condition" case; rule 1 generates an answer set for each of the three modalities (**delay, add_delay, ignore**). Rule 2 adds the temporal constraints for the cases **delay** and **add_delay**, allowing the execution of `A` after the end of the condition (and until an upper bound for the time scale). Rule 3 blocks the enforcement of temporal constraints provided in the action description (see the temporal constraints component). For the ignore modality nothing has to be changed.

```
1: 1{do_not(Tca,ID,A,delay;add_delay;ignore_delay)}1:-
       prescribeDelayWhileC(ID,C,A,T),candidate(Src,A,Tca,T).
2: tc(cg,T,A,started,Te,M):-1{do_not(T,ID,A,add_delay;delay)}1,
       prescribeDelayWhileC(ID,C,A,T),max(M),holds(C,Ts,Te),T>=Ts,T<=Te.
3: exception(cg,A,Tca,changeWFtc):-do_not(Tca,ID,A,delay).
```

The case of BMK rules prescribing the introduction of actions is similar: if `prescribe(ID,A,T)` becomes true, an answer set is generated for each execution modality reproducing the different behaviours: e.g. the **concurrent** modality enables the prescribed action, the **abort** modality enables the action and blocks all the CG candidate actions, and so on.

The **annotation rules component** consists of rules which identify discrepancies between the log trace and the different behaviours considered in the answer sets. The discrepancies are violations of the execution model, e.g. an action which is executed and not prescribed is detected by:

```
discr(action_executed_not_candidate,A,Tact):-
    started(A,Tact), not candidate(src,A,Tca,Tact).
```

and this violates formula (1a) in Sect. 3. An action executed too late, also violating (1a) because temporal constraints are not respected, is detected by:

```
discr(late,A,Tca,Tact,TcS,TcE):-candidate(Src,A,Tca,Tact),
    started(A,Tact),tc(Src,Tca,A,started,TcS,TcE),Tact>TcE.
```

ASP optimization statements are used to select the answer set with the smallest number of discrepancies, and, secondarily, to prefer conformance with more knowledge (e.g., instances of `do_not(_,_,_,_,addDelay)` are maximized).

In the example, R2 is triggered and three answer sets are generated. Some facts for each of them are shown below; e.g., in the 2nd and 3rd one (where the CG constraint is enforced) `tc(cg,55,mobilization,started,79,103)` is made true by the temporal constraints component, given that surgery ends at time 55 and mobilization has to be started 24–48 h after it. The other instance of `tc` in the 1st and 2nd answer set is due to the BMK component. Note that the two intervals have no intersection, then the solver selects the first solution, which has no discrepancies. In case of compatible constraints, if the action was started consistently with both of them, the solution with `add_delay` would be preferred.

```
1: do_not(55,r2,mobilization,delay),
   tc(cg,55,mobilization,started,140,10000),
   candidate(cg,mobilization,55,55..146)
2: do_not(55,r2,mobilization,add_delay),
   tc(cg,55,mobilization,started,140,10000),
   tc(cg,55,mobilization,started,79,103),
   discr(late,mobilization,55,146,79,103),
   candidate(cg,mobilization,55,55..146)
3: do_not(55,r2,mobilization,ignore),
   tc(cg,55,mobilization,started,79,103),
   discr(late,mobilization,55,146,79,103),
   candidate(cg,mobilization,55..146).
```

5 Conclusions and Related Work

CGs do not include all knowledge that physicians have to take into account when treating patients, since patient states and contexts of execution cannot always be foreseen in the definition of the guideline. In this paper we propose an approach for analyzing temporal conformance of execution traces with respect to a richer form of medical knowledge, which may be used to justify deviations from a strict application of the guideline, both as regards extra actions for situations that are not foreseen (and whose treatment may alter the timing of guideline execution), and cancellation or delay of actions that are prescribed by the guideline, when there are reasons to do so. Given that we do not assume

that all exceptions and interactions are modeled, interpretations provided by the approach can only be an approximation of medically correct executions. For example, in the absence of specific knowledge, the approach assumes that a temporally non-conformant execution of a guideline for treating a patient condition is always potentially justified in case another treatment, triggered by the BMK, is being executed.

Several approaches in the literature have addressed some of the issues in our proposal, or related ones. One of such issues is the verification of properties of clinical guidelines, i.e., proving that some properties hold for all executions of a guideline, in order to improve quality of guidelines. In the Protocure project, theorem proving techniques have been adopted for verification [15]: a medical protocol is modeled in the Asbru language and mapped to a specification in KIV, an interactive theorem prover. Properties are expressed in a variant of Interval Temporal Logic. Model checking techniques have been proposed in the Protocure II and GLARE projects [2,4]. Guidelines are automatically translated into a language for formal verification (SVN in Protocure II, Promela in GLARE), properties are specified in a temporal logic (ACTL and LTL respectively) and verified through model checking engines (SVN and SPIN respectively).

The integration of CGs with general medical knowledge has been considered in some case (see e.g. [16]) using such knowledge as a source of definitions of clinical terms and abstractions. In Medintel [5] different medical information sources (e.g., guidelines, reference texts, scientific literature) are used to improve decision support and the quality of care provided by general practitioners. The approach in [11] considers different forms of general medical knowledge in the context of CG verification: knowledge on the physiological mechanisms underlying the disease, and knowledge concerning good practice in treatment selection (leading to quality requirements), assuming the availability of a preference relation between treatments. Verification is used in order to check whether, given a class of patients, and considering physiological knowledge, the CG achieves a set of intentions, satisfying the quality requirements.

A limited number of approaches have dealt with verifying conformance of a trace of actions with recommendations in a CG. In [10], differences between actual actions and CG prescriptions are detected and analyzed, e.g. by comparing, for a non-compliant actions, actual findings with findings that support the action according to the CG. However, neither general medical knowledge nor quantitative time are considered in such a work. On the other hand, [3] focuses on the interaction between clinical guidelines (CGs) and the basic medical knowledge (BMK) in the light of the conformance problem. Our approach presents several similarities to it, but it is based on ASP, and, more importantly, it does not assume a fixed model for CG-BMK interactions. Indeed, different forms of interactions may be pointed out by our analysis, grounded on the different modalities we have identified. Also, the approach in this paper is an in-depth extension of the proposal in [14], addressing the temporal dimension.

Finally, our proposal is related to work about the treatment of CG *exceptions* [9,12,13], i.e., conditions that may suddenly arise, and whose treatment

is not directly foreseen in the CGs. Indeed, part of the BMK we consider in our approach regards such conditions, and some of the modalities for CG-BMK interactions have already been identified in the context of exception handling.

Acknowledgement. The authors are grateful to Gianpaolo Molino for his valuable advice on the medical issues in the paper.

References

1. Anselma, L., Terenziani, P., Montani, S., Bottrighi, A.: Towards a comprehensive treatment of repetitions, periodicity and temporal constraints in clinical guidelines. Artif. Intell. Med. **38**(2), 171–195 (2006)
2. 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)
3. Bottrighi, A., Chesani, F., Mello, P., Montali, M., Montani, S., Terenziani, P.: Conformance checking of executed clinical guidelines in presence of basic medical knowledge. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011, Part II. LNBIP, vol. 100, pp. 200–211. Springer, Heidelberg (2012)
4. 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), 1–19 (2010)
5. Brandhorst, C.J., Sent, D., Stegwee, R.A., van Dijk, B.M.A.G.: Medintel: decision support for general practitioners: a case study. In: MIE 2009, pp. 688–692 (2009)
6. British National Institute for Health and Care Excellence. Hip fracture: The management of hip fracture in adults. http://www.nice.org.uk/guidance/cg124
7. Field, M., Lohr, K. (eds.): Guidelines for Clinical Practice: From Development to Use. National Academy Press, Washington, D.C (1992). Institute of Medicine
8. Gelfond, M.: Answer sets. In: van Harmelen, F., Lifschitz, V., Porter, B. (eds.) Handbook of Knowledge Representation, Chap. 7. Elsevier, Amsterdam (2007)
9. Grando, A., Peleg, M., Glasspool, D.: A goal-oriented framework for specifying clinical guidelines and handling medical errors. J. Biomed. Inform. **43**(2), 287–299 (2010)
10. Groot, P., Hommersom, A., Lucas, P.J.F., Merk, R., ten Teije, A., van Harmelen, F., Serban, R.: Using model checking for critiquing based on clinical guidelines. Artif. Intell. Med. **46**(1), 19–36 (2009)
11. Hommersom, A., Groot, P., Lucas, P.J.F., Balser, M., Schmitt, J.: Verification of medical guidelines using background knowledge in task networks. IEEE Trans. Knowl. Data Eng. **19**(6), 832–846 (2007)
12. Leonardi, G., Bottrighi, A., Galliani, G., Terenziani, P., Messina, A., Della Corte, F.: Exceptions handling within GLARE clinical guideline framework. In: AMIA 2012, American Medical Informatics Association Annual Symposium (2012)
13. Quaglini, S., Stefanelli, M., Lanzola, G., Caporusso, V., Panzarasa, S.: Flexible guideline-based patient careflow systems. Artif. Intell. Med. **22**(1), 65–80 (2001)
14. Spiotta, M., Bottrighi, A., Giordano, L., Dupré, D.T.: Conformance analysis of the execution of clinical guidelines with basic medical knowledge and clinical terminology. In: Miksch, S., Riano, D., Teije, A. (eds.) KR4HC 2014. LNCS, vol. 8903, pp. 62–77. Springer, Heidelberg (2014)

15. ten Teije, A., Marcos, M., Balser, M., van Croonenborg, J., Duelli, C., van Harmelen, F., Lucas, P.J.F., Miksch, S., Reif, W., Rosenbrand, K., Seyfang, A.: Improving medical protocols by formal methods. Artif. Intell. Med. **36**(3), 193–209 (2006)
16. ten Teije, A., Miksch, S., Lucas, P. (eds.): Computer-based Medical Guidelines and Protocols: A Primer and Current Trends, Studies in Health Technology and Informatics, vol. 139. IOS Press, Amsterdam (2008)
17. Terenziani, P., Molino, G., Torchio, M.: A modular approach for representing and executing clinical guidelines. Artif. Intell. Med. **23**(3), 249–276 (2001)
18. The Potsdam Answer Set Solving Collection. http://potassco.sourceforge.net/

Towards a Pathway-Based Clinical Cancer Registration in Hospital Information Systems

Michael Heß[1(✉)], Monika Kaczmarek[1], Ulrich Frank[1], Lars-Erik Podleska[2], and Georg Taeger[2]

[1] Research Group Information Systems and Enterprise Modeling, University of Duisburg-Essen, Essen, Germany
{m.hess,monika.kaczmarek,ulrich.frank}@uni-due.de

[2] Division of Sarcoma Surgery, Department of General-, Visceral- and Transplant Surgery, West German Cancer Center, University Hospital, University of Duisburg-Essen, Essen, Germany
{lars.podleska,georg.taeger}@uk-essen.de

Abstract. The clinical cancer registration constitutes a major source of data used to support the cancer research. However, providing data to cancer registries usually requires additional effort, as the automated extraction of the required data from the already existing clinical documentation is often not possible. In this paper, we show how the conceptual models of clinical pathways (CPs), created using the domain-specific modeling language extended with the data models reconstructed based on the common oncological basic data set, can support a model-based design and development of process-oriented hospital information systems and contribute to enhancing the data-quality and efficiency of clinical cancer registration.

Keywords: Multi-Perspective Hospital Modeling · DSML · Cancer registration

1 Introduction

Hospitals are complex socio-technical systems with a high number of different professions and disciplines [14, p. 119]. They are to deliver personalized, high-quality and cost-effective medical care [37]. Thus, a paradigm shift from the function-oriented to process-oriented delivery of medical care took place [44, p. 1]. As a consequence, the widely accepted concepts of clinical practice guidelines [17] and clinical pathways (CPs) [13] have been introduced in the majority of hospitals in order to increase the quality of care and its cost-effectiveness [39]. This shift affected also hospital information systems (HIS) [37,47] which needed to be aligned with the process-oriented care delivery in order to better support medical staff, e.g., by reducing administrative workload, such as documentation, patient scheduling, or material management, and by providing all required information at the point and time of care.

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 80–94, 2015.
DOI: 10.1007/978-3-319-26585-8_6

A process-oriented HIS "formally models [and executes] guidelines, workflows, or care pathways and provides support for clinical decisions that extend over time" [22, p. 739]. Its development is "an iterative, collaborative process that involves defining a clinical process model comprising formalized medical knowledge [...] and organizational workflow" [22, p. 744]. One of the critical success factors of HIS development is to involve domain experts and prospective users already in the design phase [4, p. 155]. In order to facilitate communication and understanding between IT experts and domain experts, quite often the conceptual models of the CPs to be supported by the system are used. The application of a domain-specific modeling language (DSML) instead of a general-purpose modeling language to create such conceptual models of CPs may offer several benefits [21, pp. 133–134], among others, higher quality and increased understandability of created models due to the higher degree of domain-specific semantics as the modeling concepts correspond with the professional terminology of the domain experts. Finally, conceptual models of CPs may be used as a foundation for a model-based design of process-oriented HIS [31,37].

Taking into account the ever increasing documentation obligations resulting out of regulatory and organizational requirements or out of scientific research [2,5], HIS, besides other features, should contribute to reducing the required documentation effort [45, p. 545]. Indeed, the daily amount of working time that physicians are spending on clinical and/or administrative documentation is reported to be up to 40.6 % [5, pp. 545 f.]. Thus, the support of an electronic and partly automated documentation is one of the key features that electronic CPs should offer [45]. Initially, this may result in additional time to be spent on documentation, however in consequence, the later efforts are reduced and/or the quality of the documentation increases [4, p. 155]. Therefore, not only the reuse of data and integration of heterogeneous parts of HIS with respect to clinical documentation is required (to avoid redundancy and multiple data entry), but also efficient ways to support additional documentation are called for [32, p. 398], [7, p. 90].

The need to support additional documentation is especially visible in the field of oncology providing highly interdisciplinary and complex medical care. Reporting each new cancer diagnose and data on the course of treatment to the clinical and from there to the population-based cancer registries is, depending on regional/national regulations, either obligatory or optional but highly recommended [29], [7, p. 90]. To support an efficient use of electronic CPs the "availability of structured data" [4, p. 151] sets as means of documentation is required. To foster comparability and exchangeability of cancer-related documentation, e.g., between care providers or for the needs of quality assurance, a commonly defined and accepted data set is demanded [8, p. 7]. The common oncological basic data set (COBDS) provides the specification of corresponding data structures [30] and constitutes the first of three levels of cancer documentation in Germany [23, p. A-1041]. It specifies a minimal data set, independent from the specific tumor entity, which needs to be recorded and delivered to clinical and population-based cancer registries.

We argue that models of CPs enhanced with corresponding documentation structures can serve as a foundation for a model-driven (re-)design of HIS [31] and contribute to enhancing the data-quality and efficiency of clinical cancer registration. Thus, our goal is to show how the existing modeling language for clinical pathways [25], being the core of the Multi-Perspective Hospital Modeling (MPHM) method [24], can be enhanced with semantically rich data models in support of the model-based (re-)design of process-oriented HIS in the context of oncology on the example of supporting clinical cancer registration.

The conducted research is part of a project aiming at the design and development of a domain-specific modeling method for hospitals [24] and follows the design-oriented information systems research [35]. The paper is structured as follows. First, the proposed Multi-Perspective Hospital Modeling method is briefly introduced and the main features of its DSML4CPs are outlined. Then, the performed data model reconstruction is discussed and its integration into the DSML4CPs is shown. Next, an exemplary application of the extended DSML4CPs is given. The paper concludes with an outlook on future research.

2 Domain-Specific Modeling Language for CPs in the Realm of Multi-Perspective Hospital Modeling

The MPHM method is a domain-specific extension of the Multi-Perspective Enterprise Modeling (MEMO) method [21] and aims at providing hospitals with an instrument, which is tailored to their specific needs and allows to model not only clinical pathways, but also other elements of the hospital's action system and information system (IS) in order to address information needs of all stakeholders. Therefore, the aim is not only to increase the transparency of undertaken actions and foster communication, but also to provide a basis for conducting purposeful model-based analyses taking into account various professional perspectives (e.g., medical, administrative, and technical) [24, pp. 372–373]. The method is also to constitute a foundation for (re-)designing the hospital's action system and IS to increase efficiency and/or effectiveness of all hospital's processes [24, pp. 372–373]. Although the proposed method aims at covering the entire field of medicine, with respect to its complexity as well as the medical and societal relevance, the medical field of oncology has been chosen as a current focus of the proposed method (for details see [25]).

The core of the MPHM method is the Domain-Specific Modeling Language for Clinical Pathways (DSML4CPs). A number of initiatives in the disciplines of Information Systems (e.g., [10,16,33]), medicine and medical informatics (e.g., [6,40,42,43]) focus on modeling CPs. The approaches differ regarding the extent of the formalization of the treatment logic and their expressiveness. In our previous work we have identified a set of requirements that should be met by the DSML4CPs and used it to evaluate the existing approaches (cf. [25]). Based on the conducted evaluation the most promising approach (taking into account the defined criteria), i.e., MEMO Organisation Modeling Language (OrgML) [18], was selected to be extended towards a DSML4CPs.

A modeling language, such as the proposed DSML4CPs, is defined through its abstract syntax and semantics (specified, e.g., in the form of a meta model, i.e., a model of models) and concrete syntax (a graphical notation) (cf. [25]). A meta model defines the abstract syntax and semantics of a given language together with additional constraints [20, p. 3]. Thus, a model (M1 level) is created using a modeling language, which in turn is specified by a meta model (M2 level). Thus, a model is an instance of a meta model, which in turn is an instance of a meta meta model (M3 level) [20, p. 3]. The constraints can be formulated using, e.g., Object Constraint Language (OCL) [46].

In this paper, we propose an enrichment of the DSML's process meta types by associating data structures corresponding to the documentation needs in the field of oncology, which may be seen as one step towards the support of a model-based (re-)design and development of a process-oriented HIS that provides semantically richer documentation structures corresponding with the clinical context. In addition, the enhanced CP models should foster model-based analyses.

In order to enhance the DSML4CPs with the corresponding documentation structures, first, if not yet available, the required data structures need to be specified/reconstructed (e.g., using the Entity Relationship Model [11]). Then, based on the data model(s), the conceptual links between DSML4CPs' meta types and the entity types can be identified. If necessary, extensions to the meta model may be specified. Finally, the meta types are linked with the entity types. This procedure was applied in our case to integrate the required data structures in support of cancer registration as shown subsequently.

3 Documentation and Data Structures for Cancer Registration

Early efforts to support clinical data management in oncology, e.g., by Altmann et al. [1], resulted in the development of a reference model for cancer documentation. The German Health Level 7 (HL7) user group published an implementation guide for the transfer of oncological data between different IS using the Clinical Document Architecture (CDA), version 2 [28]. The HL7 CDA aims at describing documents in the form of XML documents consisting out of elements and attributes in order to take advantage of a structured representation which is both human-understandable and machine-interpretable [3, pp. 159 ff.], [12, p. 85]. Finally, the working group of the German Association of Comprehensive Cancer Centers (ADT) and the Association of Population-based Cancer Registries in Germany (GEKID) have consented and published the specification of the common oncological basic data set [30] (COBDS) that specifies structures for data that have to be reported on each patient's cancer disease to the clinical and later on population-based cancer registry. Its application is obligatory for German healthcare institutions treating patients with cancer diseases to allow for the nationwide data-exchange and analyses in corresponding clinical and population-based registries [9].

Taking into account our goal, we focus on the COBDS as, on the one hand, it offers the basic structure that allows for later extensions, e.g., to support tumor

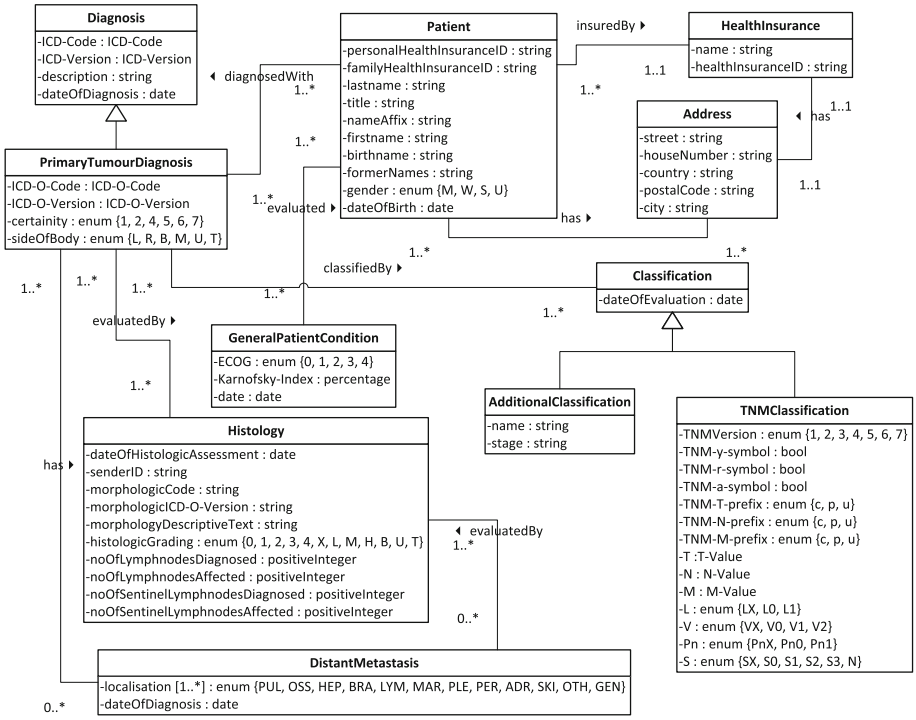


Fig. 1. Data model – Part 1: Patient and Diagnostic Data

entity specific cancer registration and, on the other hand, due to its generality, it is applicable to all tumor entities. To the best of our knowledge no (domain-specific) modeling language supporting modeling of CPs exists that specifies data structures corresponding to the COBDS. This seems to be because of two reasons: firstly, approaches focusing on modeling CP with the aim to support process execution do not aim at a model-based (re-)design of HIS as such. Secondly, up till now no other DSML providing comparably detailed and differentiated concepts to model oncological pathways has been proposed – instead they provide rather generic modeling concepts, such as process/action/plan, event/(clinical) state, and decision/decision scenario (e.g., [10,16,33,36,40]).

The COBDS's XML specification has been reconstructed in the form of entity relationship diagrams (cf. Figs. 1 and 2). All entity types are detailed with a number of attributes, the majority of which is of type enumeration. In case an enumeration allows to document more than one value, the cardinality is explicitly indicated as [1..*], e.g., in the entity type `SystemicTherapy`.[1]

Figure 1 focuses on entity types with respect to modeling the patient's master data (`Patient, Address`) and diagnostic data, such as the patient's tumor

[1] Entity types reconstructed from COBDS are typeset in `typewriter font` whereas concepts specified in the DSML4CPs are typeset in *`typewriter italic font`*.

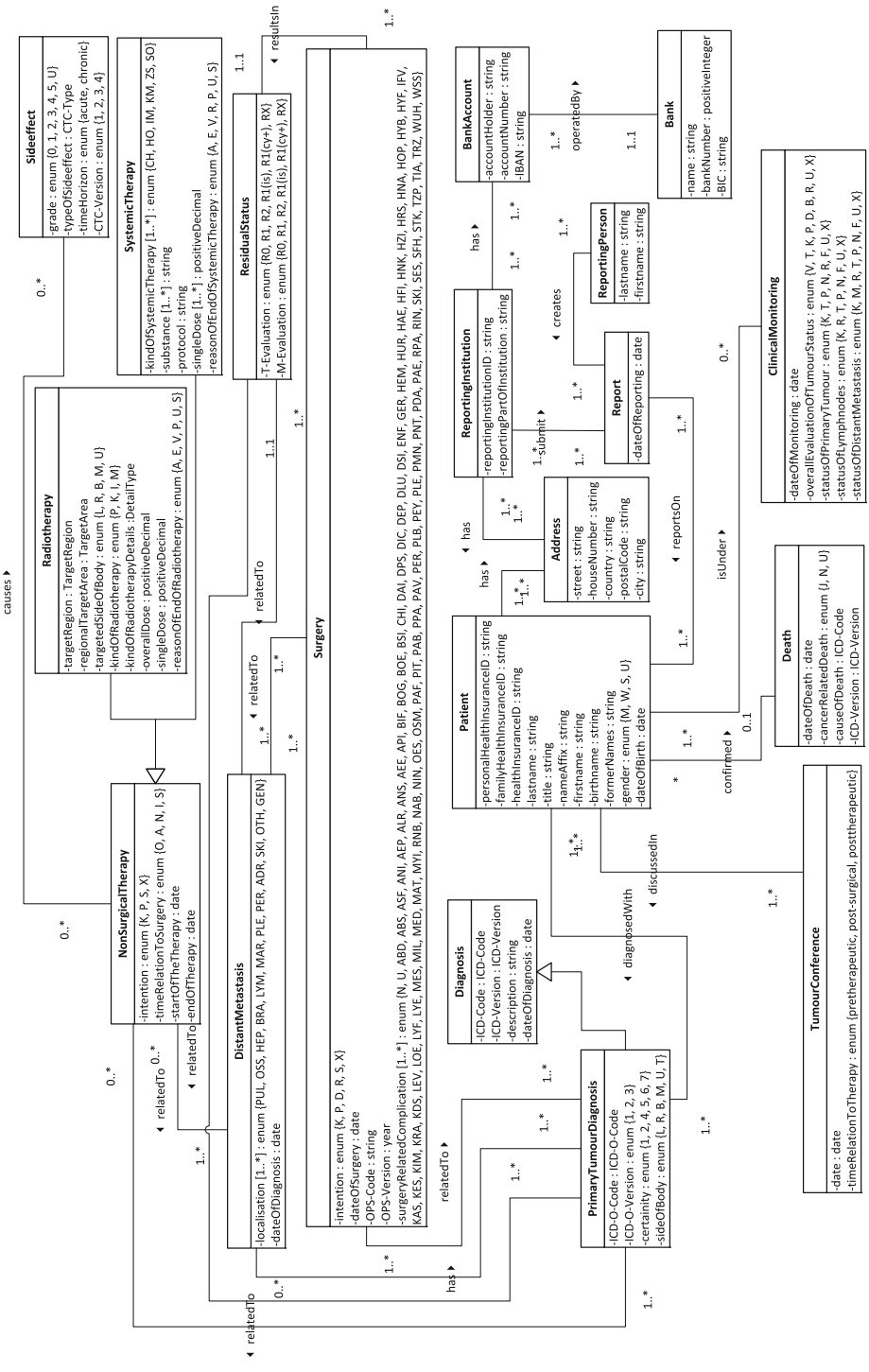


Fig. 2. Data model – Part 2: Patient, Therapeutic and Follow-Up Data

diagnose including potential metastasis/es (`Diagnosis` and `TumourDiagnosis`, and `DistantMetastasis`), general condition (`GeneralCondition`), results of histologic examination (`Histology`) as well as of the tumor staging (entity type `Classification`, specialized into the `TNMClassification` applied to stage most tumor entities, and `AdditionalClassification` to allow for documenting the result of the tumor staging according to some other classification) at the end of the diagnostics. Furthermore, an entity type documenting information on the patient's health insurance (`HealthInsurance`) is provided.[2]

Figure 2 provides entity types focusing on the documentation related to the therapeutic process types and to the follow-up, once the course of treatment has ended.[3] With respect to the documentation of the main options of therapeutic process types in oncology, the entity types `Radiotherapy`, `Surgery`, and `SystemicTherapy` have been specified. As `Radiotherapy` and `SystemicTherapy` share common attributes, out of them the entity type `NonSurgicalTherapy` has been generalized. To allow for documentation of side effects during radiotherapy and systemic therapy, the corresponding entity type `SideEffect` is specified and associated with the entity type `NonSurgicalTherapy`. All therapeutic modalities can be performed to treat both, the primary tumor and one or more potentially detected metastasis/es. Metastases can be documented using the entity type `DistantMetastasis`. After a surgery, the tumor's/metastases' residual status (`ResidualStatus`) needs to be documented. Furthermore, each discussion on a patient's case conducted during the interdisciplinary tumor conferences needs to be documented (`TumourConference`). Once the course of treatment ended, each patient undergoes a continuous follow-up. For documenting the results of each following check, the entity type `ClinicalMonitoring` is provided. In case the patient does not survive the cancer disease, the death needs to be documented as well (`Death`). As the reporting institution (in Germany) gets a financial re-compensation for reporting to cancer registries, the institution's master data (`ReportingInstitution`, `Address` and `BankAccount`) and, in case further inquiries occur, the reporting person (`ReportingPerson`), need to be documented.

To provide a better understanding of the reconstructed data model, Table 1 provides an exemplary overview of the meaning of the permitted values of the enumeration-attribute 'ECOG' (Eastern Cooperative Oncology Group) patient performance status [34] of the entity type `GenenralPatientCondition`. Explanations of the meaning of all other permitted values in Figs. 1 and 2 can be found in the specification of the COBDS [30].

[2] Please note that the constraint to ensure that the patient's address is different from the address of the health insurance is omitted.

[3] For the needs of readability, the reconstructed data model is presented in two parts, which is why the entity types `Patient`, `Diagnosis`, `TumourDiagnosis`, `DistantMetastasis`, and `Address` are part of Figs. 1 and 2.

Table 1. Possible values of the enum.-attr. 'ECOG' of the entity `ClinicalMonitoring`

Attr	Value	Meaning
ECOG	0	Fully active, able to carry on all pre-disease performance without restriction
	1	Restricted in physically strenuous activity but ambulatory and able to carry out work of a light or sedentary nature, e.g., light house work, office work
	2	Ambulatory and capable of all selfcare but unable to any work activities. Up and about more than 50 % of waking hours
	3	Capable of only limited selfcare, confined to bed/chair more than 50 % of waking hrs
	4	Completely disabled. Cannot carry on any selfcare. Totally confined to bed or chair
	5	Dead

4 Integration of Data Structures into the DSML4CPs' Specification

Defining a meta model requires making a number of modeling decisions, e.g., should a concept be part of the language specification (i.e., a meta type, M2 level) or part of the language application (i.e., a type, M1 level). In addition, the decisions regarding attributes, their types and relations between concepts and their cardinalities need to be made. To support making design decisions and to ensure the required quality of the developed meta model, the guidelines proposed by [19,21] were applied. If the application of the above guidelines did not lead to the desired effect, the classification of a concept as a local type [19], application of intrinsic features (marked by the literal "i" in white color on a black background) indicating that a meta type or meta attribute can be instantiated only on M0 level [19, p. 104] and language level types (visualized with a black name of the concept on a grey background) allowing for specifying concepts that represent instances already on the type level [20, pp. 23–24] were considered.

Figure 3 shows an excerpt of the meta model of the DSML4CPs extended with entity types reconstructed from the COBDS, presented in the previous section. The association of the entity types to selected process meta types has been straight possible, as some of the DSML's process meta types have been among others partly specified based on the COBDS. Candidates for the DSML's process meta types have been identified based on a literature study, interviews with domain experts working at a Comprehensive Cancer Center and observations of their daily working routines. For a detailed description of the requirements analysis process, the DSML's design, and design decisions, cf. [25]. The DSML4CPs is specified as an extension of the MEMO Organisation Modeling Language (OrgML [18]). Concepts originating from MEMO OrgML are indicated in the meta model by a quadratic icon filled with diagonal lines. As the basic abstraction, the DSML4CPs provides the process meta types

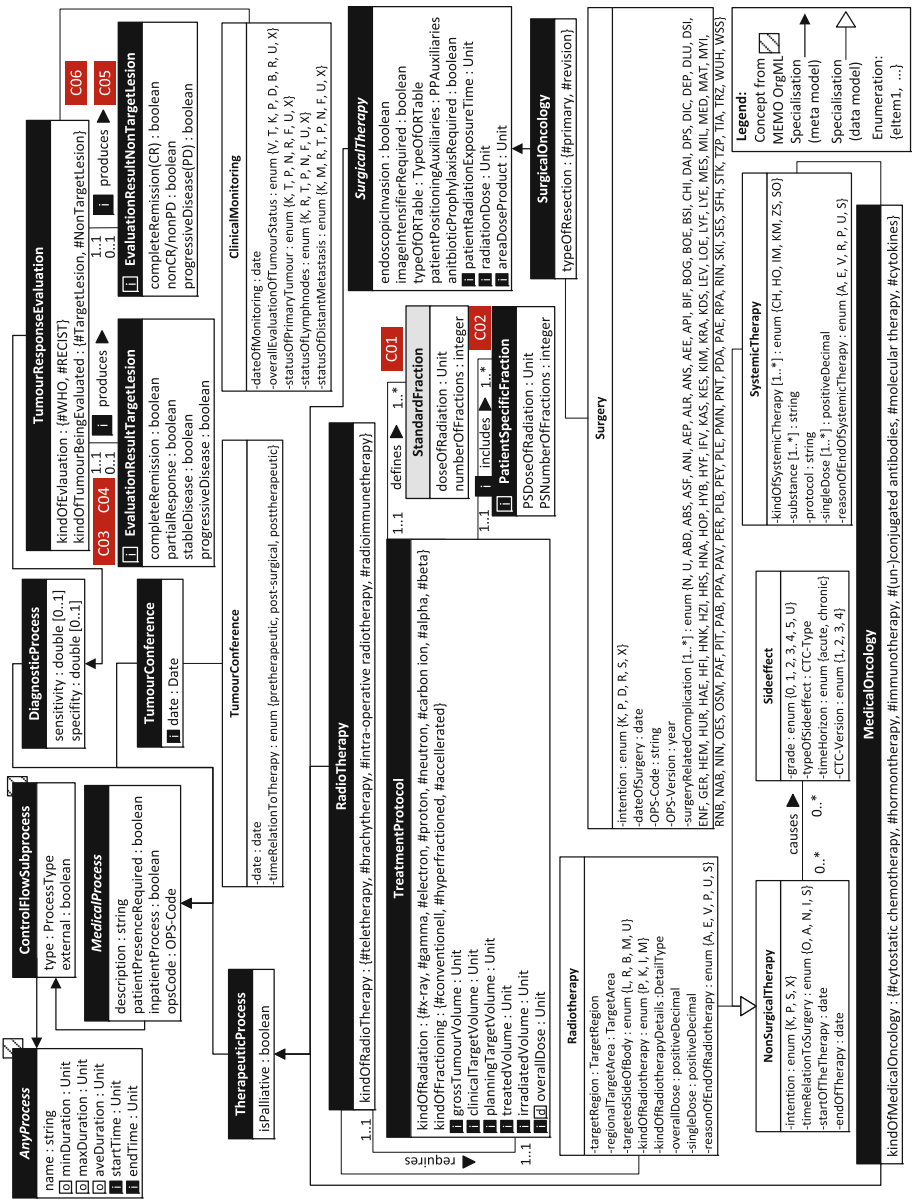


Fig. 3. Meta model excerpt of the DSML for CPs [25] integrated with data structures reconstructed from the common oncological basic data set.

DiagnosticProcess and *TherapeuticProcess*, which represent the basic distinction of medical care process types. Common attributes are specified in the generalized, abstract process meta type *MedicalProcess*. To allow for creating CP models with a more detailed and semantically richer specialization of medical processes and thereby providing a semantically richer representation of CP, *DiagnosticProcess* and *TherapeuticProcess* have been specialized into several more process meta types. In case of *DiagnosticProcess*, e.g., the concepts *DiagnosticImaging*, *LaboratoryExamination* (both not included in this excerpt to preserve the figure's readability), and *TumourResponseEvaluation*, which supports modeling of evaluation of target lesions and non target lesions, have been specified. In case of *TherapeuticProcess*, e.g., the concepts *MedicalOncology*, *RadioTherapy*, and *SurgicalTherapy* have been specified. These concepts constitute abstractions over the three main pillars of oncological therapy modalities [41, p. 89] and directly correspond to the specific partial data sets of the COBDS.

Consequently, the entity types are associated with the process meta types as follows: The entity types `SystemicTherapy`, `Radiotherapy`, and `Surgery` are associated with the specialized therapeutic process meta types *MedicalOncology*, *RadioTherapy*, and *SurgicalOncology*. Furthermore, the entity type `TumourConference` is associated with the process meta type *TumourConference* and the entity type `ClinicalMonitoring` with the process meta type *TumourResponseEvaluation*. All potential values that may be documented by an enumeration are taken directly from the source of the reconstruction, i.e., the COBDS.

The meta model contains additional constraints formulated using the Object Constraint Language (OCL) to foster the integrity and semantic correctness of the models being created on the M1 level. For details see [25].

5 Exemplary Application

In the following, we point to three application scenarios of the proposed extensions indicating the data reuse potential and model-based analysis possibilities. Regarding the data reuse potential, Fig. 4 shows in the upper part a generic, high-level abstraction over the oncological clinical pathways. The following, general steps can be distinguished: *administrative and medical patient admission*, *local and spread diagnostics* leading to the *decision for a therapeutic regime*, followed by the actual *therapeutic procedures*. Once the course of treatment ends, patients will undergo a continuous *follow-up*. In case of the death of a patient, optionally an *autopsy* may be performed. Below each generic step of the CP, corresponding data items for the needs of documentation are shown (light grey boxes). The dashed arrows indicate which data items from which step of the CP have to be documented in which part of the COBDS. The potential for reuse of data initially entered into the clinical documentation for the needs of cancer registration becomes apparent. To realize this potential, adequate interfaces and data-exchange standards need to be specified, for instance, using HL7 CDA.

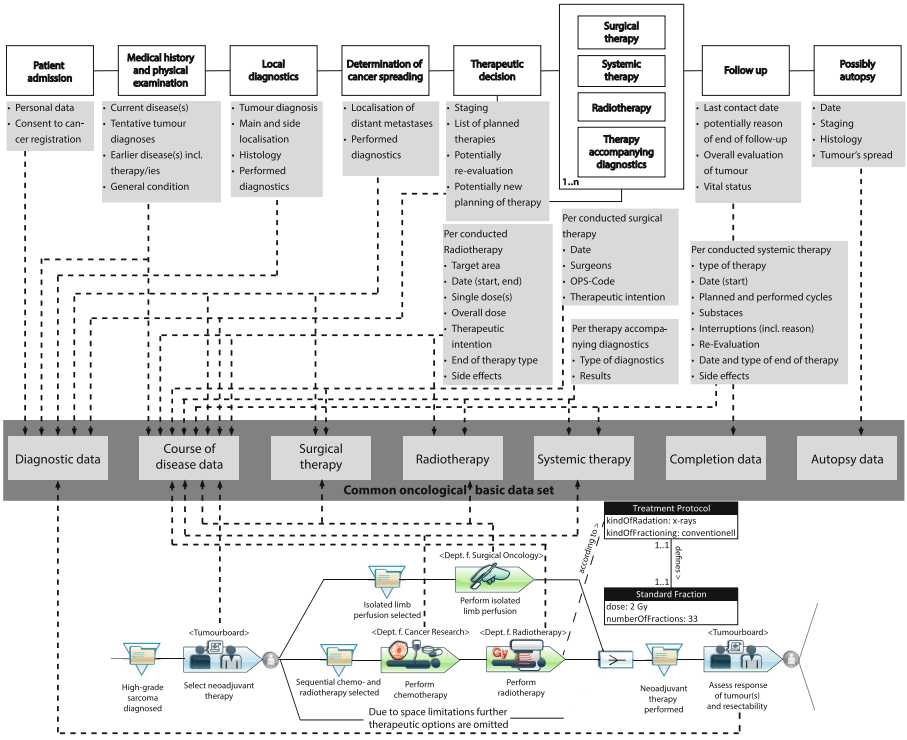


Fig. 4. COBDS-conform documentation needs of an oncologic CP (cf. [25,26]).

In the lower part of the figure, an excerpt of the CP *Diagnostics and therapeutic strategy for Soft Tissue Sarcoma* modeled with the DSML4CPs is shown (for an extended version of the CP that has been modeled together with the West German Sarcoma Center cf. [25]). Based on the proposed extension of the meta model (cf. Fig. 3), in the resulting diagram again dashed arrows indicate which data items from which steps of the now more detailed CP have to be documented in which part of the COBDS. As can be seen on the example of the simplified radiotherapeutic treatment protocol, the visualization of patient-, i.e., instance-specific data in a CP's diagram may provide a quite intuitive and focused representation which might allow for providing a comprehensive overview of the patient's individual documentation related to the CP.

The documented patient-specific data can also be used to support decision making, e.g., for identifying the most promising course of treatment. Criteria and rules a decision is based on can be stored in the model, e.g., using decision tables [27,38]. Conditions and values can be transformed into rules that can be inferred by knowledge-based expert system components of process-aware HIS.

Furthermore, a graphical representation of process models, i.e., here CP models, may serve as an interface for visualizing and 'navigating' through not only case-specific but also aggregated data of a number of executed instances of a CP.

Such an interface – together with a corresponding underlying implementation of analysis mechanisms – might enable physicians to perform selected analyses based on the CP model without having to involve medical controllers or IT experts, by selecting a step of the CP and corresponding data of interest. This might be starting from demographic data, the distribution of tumor locations and staging according to the patients' age and/or gender. With respect to quality of care assurance, the patients' outcome might be analyzed in relation to the selected therapeutic regime, or organizational aspects, such as the timespan from the initial consultation of a specialist to the point of time of the patient's discussion at the interdisciplinary tumor conference for deciding on the therapeutic strategy.

6 Conclusions

In this paper, we have shown how the already developed DSML4CPs can be enhanced with the data structures relevant to cancer registration in support of model-based HIS (re-)design and indicated its possible impact on the functionality of resulting HIS. However, the presented DSML allows to create CP models only on the type level, i.e., it does not offer a process description detailed enough to generate detailed, executable workflow schemata [15, p. 42]. Nevertheless, enriching the DSML's process meta types by associating data structures corresponding to the documentation needs of the specific medical field, may be seen as one step towards the support of a model-based (re-)design and development of a process-oriented HIS that provides adequate documentation structures corresponding with the clinical context. Finally, in line with the goals of MPHM, the enhanced CP models foster model-based analyses.

The DSML4CPs has been designed with a focus on oncology. Consequently, the presented extension with data structures focuses on cancer registration, which may serve as a starting point for a more comprehensive enhancement of the DSML's specification with respect to general clinical documentation in oncology as well as for an extended tumor entity-specific documentation, e.g., in support of more detailed analyses for needs of cancer research.

References

1. Altmann, U., Katz, F.R., Dudeck, J.: A reference model for clinical tumour documentation. In: Hasman, A., Haux, R., et al. (eds.) Ubiquity: Technologies for Better Health in Aging Societies, pp. 139–144. IOS Press, Maastricht (2006)
2. Ammenwerth, E., Spötl, H.P.: The time needed for clinical documentation versus direct patient care. Meth. Inf. Med. **48**(1), 84–91 (2009)
3. Benson, T.: Principles of Health Interoperability HL7 and SNOMED, 2nd edn. Springer, London (2012)
4. Blaser, R., Schnabel, M., et al.: Improving pathway compliance and clinician performance by using information technology. IMJI **76**(2–3), 151–156 (2007)
5. Blum, K., Müller, U.: Dokumentationsaufwand im Ärztlichen Dienst der Krankenhäuser. Das Krankenhaus **95**(7), 544–548 (2003)

6. Boxwala, A.A., et al.: GLIF3: a representation format for sharable computer-interpretable clinical practice guidelines. J. Biomed. Inform. **37**(3), 147–161 (2004)
7. Breil, B., Dugas, M.: Transferring HIS data to population-based cancer registries. In: Adlassnig, K.P., Blobel, B., Mantas, J., Masic, I. (eds.) Medical Informatics in a United and Healthy Europe, pp. 86–90. IOS Press, Sarajevo (2009)
8. Bundesministerium für Gesundheit: Nationaler Krebsplan. Handlungsfelder, Ziele und Umsetzungsempfehlungen, Berlin (2012)
9. Bundesministerium für Gesundheit: Aktualisierter einheitlicher onkologischer Basisdatensatz der Arbeitsgemeinschaft Deutscher Tumorzentren e. V. (ADT) und der Gesellschaft der epidemiologischen Krebsregister in Deutschland e. V. (GEKID). Bundesanzeiger vom April 28, 2014 (2014)
10. Burwitz, M., Schlieter, H., Esswein, W.: Modeling clinical pathways - design and application of a domain-specific modeling language. In: Alt, R., Franczyk, B. (eds.) Proceedings of Wirtschaftsinformatik 2013, vol. 2, pp. 1325–1339, Leipzig (2013)
11. Chen, P.P.S.: The entity relationship model - toward a unified view of data. ACM Trans. Database Syst. **1**(1), 9–36 (1976)
12. Cuggia, M., Avillach, P., Daniel, C.: Representation of patient data in health information systems and electronic health records. In: Venot, A., Burgun, A., Quantin, C. (eds.) Medical Informatics, e-Health, pp. 65–89. Springer, Paris (2014)
13. de Bleser, L., Depreitere, R., Waele, K.D., Vanhaecht, K., Vlayen, J., Sermeus, W.: Defining pathways. J. Nurs. Manage. **14**(7), 553–563 (2006)
14. Drucker, P.F.: Managing in the Next Society. Butterworth-Heinemann, Oxford (2002)
15. Dumas, M., Rosa, M.L., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Heidelberg (2013)
16. Färber, M., Jablonski, S., Schneider, T.: A Comprehensive Modeling Language for Clinical Processes. In: Hein, A., Thoben, W., Appelrath, H.J., Jensch, P. (eds.) European Conference on eHealth 2007, Proceedings of the ECEH 2007, pp. 77–88. Gesellschaft für Informatik e. V., Oldenburg (2007)
17. Field, M.J., Lohr, K.N.: Clinical Practice Guidelines: Directions for a New Program. The National Academies Press, Washington (DC) (1990)
18. Frank, U.: Memo organisation modelling language (2): Focus on business processes. ICB Research Report 49, University of Duisburg-Essen (2011)
19. Frank, U.: Some guidelines for the conception of domain-specific modelling languages. In: Nüttgens, M., Thomas, O., Weber, B. (eds.) Proceedings of EMISA 4th International Workshop. LNI, vol. 190, pp. 93–106. GI, Bonn (2011)
20. Frank, U.: The MEMO Meta Modelling Language (MML) and Language Architecture. 2nd edn. ICB-Research Report 43, University of Duisburg-Essen (2011)
21. Frank, U.: Domain-specific modeling languages: requirements analysis and design guidelines. In: Reinhartz-Berger, I., Sturm, A., Clark, T., Cohen, S., Bettin, J. (eds.) Domain Engineering, pp. 133–157. Springer, Berlin (2013)
22. Gooch, P., Roudsari, A.: Computerization of workflows, guidelines, and care pathways: a review of implementation challenges for process-oriented health information systems. JAMIA **18**, 738–748 (2011)
23. Haier, J.: Datenmanagement in integrierten Krebszentren: Harmonisierung der Strukturen erforderlich. Deutsches Ärzteblatt **106**(21), A-1040–A-1043 (2009)
24. Heß, M.: Towards a domain-specific method for multi-perspective hospital modelling – motivation and requirements. In: vom Brocke, J., Hekkala, R., Ram, S., Rossi, M. (eds.) DESRIST 2013. LNCS, vol. 7939, pp. 369–385. Springer, Heidelberg (2013)

25. Heß, M., Kaczmarek, M., Frank, U., Podleska, L., Taeger, G.: Towards a DSML for clinical pathways in the realm of multi-perspective hospital modelling. In: Becker, J., vom Brocke, J., de Marco, M. (eds.) Proceedings of 23rd ECIS, Münster (2015)
26. Heß, M., Podleska, L., Täger, G.: Vision einer integrierten Tumordokumentation (Poster). In: 20 Informationstagung Tumordokumentation, Lübeck (2013)
27. Heß, M., Schlieter, H., Täger, G.: Modellierung komplexer Entscheidungssituationen in Prozessmodellen - Anwendung am Beispiel der Tumorklassifikation bei Weichteilsarkomen. In: Thomas, O., Nüttgens, M. (eds.) Dienstleistungsmodellierung 2012, pp. 268–290. Springer Gabler, Wiesbaden (2012)
28. HL7-Benutzergruppe in Deutschland e. V.: Implementierungsleitfaden Übermittlung von onkologischen Daten mittels HL7 CDA R2. Technical report, Köln (2011)
29. Katalinic, A., Richter, A., et al.: Krebsregistrierung im europäischen und nichteuropäischen Ausland. Der Onkologe **19**(12), 1025–1036 (2013)
30. Klinkhammer-Schalke, M., Hentschel, S., et al.: Einheitlicher onkologischer Basisdatensatz der ADT/GEKID (Stand: 12.02.2014). ADT und GEKID (2014)
31. Krogstie, J.: Model-Based Development and Evolution of Information Systems. A Quality Approach. Springer, London (2012)
32. Lenz, R., Blaser, R., et al.: IT support for clinical pathways - lessons learned. Int. J. Med. Inform. **76**(Suppl. 3), S397–S402 (2007)
33. Meiler, C.: Modellierung Planung und Ausführung Klinischer Pfade. ibidem-Verlag, Stuttgart (2005)
34. Oken, M.M., Creech, R.H., Torney, D.C., Horton, J., Davis, T.E., McFadden, E.T., Carbone, P.P.: Toxicity and response criteria of the eastern cooperative oncology group. Am. J. Clin. Oncol. **5**, 649–655 (1982)
35. Österle, H., Becker, J., Frank, U., et al.: Memorandum on design-oriented information systems research. EJIS **20**, 7–10 (2011)
36. Peleg, M.: Computer-interpretable clinical guidelines: a methodological review. J. Biomed. Inform. **46**(4), 744–763 (2013)
37. Reichert, M.: What BPM technology can do for healthcare process support. In: Peleg, M., Lavrač, N., Combi, C. (eds.) AIME 2011. LNCS, vol. 6747, pp. 2–13. Springer, Heidelberg (2011)
38. Riaño, D.: A systematic analysis of medical decisions: how to store knowledge and experience in decision tables. In: Riaño, D., ten Teije, A., Miksch, S. (eds.) KR4HC 2011. LNCS, vol. 6924, pp. 23–36. Springer, Heidelberg (2012)
39. Rotter, T., Kinsman, L., et al.: Clinical pathways: effects on professional practice, patient outcomes, length of stay and hospital costs (review). Cochrane Database Syst. Rev., **7**(3) (2010)
40. Med. Decis. Making, Society for medical decision making: proposal for clinical algorithm standards. **12**(2), 149–154 (1992)
41. Stephens, F.O., Aigner, K.R.: Basics of Oncology. Springer, Berlin (2009)
42. Sutton, D., Taylor, P., Earle, K.: Evaluation of proforma as a language for implementing medical guidelines in a practical context. BMC Med. Inform. Decis. Making **6**, 1–11 (2006)
43. Tu, S., Campbell, J.R., Glasgow, J., Nyman, M.A., McClure, R., et al.: The sage guideline model: achievements and overview. JAMIA **14**(5), 589–598 (2007)
44. Vos, L., et al.: Towards an organisation-wide process-oriented organisation of care: a literature review. Implementation Sci. **6**(8), 1–14 (2011)
45. Wakamiya, S., Yamauchi, K.: What are the standard functions of electronic clinical pathways? Int. J. Med. Inform. **78**(8), 543–550 (2009)

46. Warmer, J.B., Kleppe, A.G.: The Object Constraint Language : Getting Your Models Ready for MDA, 2nd edn. Addison-Wesley, Boston (2003)
47. Winter, A., Haux, R., Ammenwerth, E., Brigl, B., Hellrung, N., Jahn, F.: Health Information Systems. Architectures and Strategies, 2nd edn. Springer, London (2011)

A Mixed-Initiative Approach to the Conciliation of Clinical Guidelines for Comorbid Patients

Luca Piovesan[1(✉)] and Paolo Terenziani[2]

[1] Department of Computer Science, Università degli Studi di Torino, Turin, Italy
piovesan@di.unito.it
[2] DISIT, Institute of Computer Science, Università del Piemonte Orientale, Alessandria, Italy
paolo.terenziani@unipmn.it

Abstract. The treatment of patients affected by multiple pathologies (comorbid patients) is one of the main challenges for the modern healthcare. Clinical practice guidelines provide evidence-based information about interventions considering only single pathologies. To support physicians in the treatment of comorbid patients, suitable methodologies must be devised. In our previous work, we have (i) devised an ontology of interactions, (ii) proposed a mixed-initiative approach (based on the ontology) to support physicians in the detection of interactions between guidelines, and (iii) extended our approach to consider the temporal dimension. In this paper, we move a step forward, by (iv) identifying a set of practical "management options" used by physicians in their daily practice to manage interactions between guidelines and (v) providing the methodologies to support physicians in the application of such management options in reconciling the guidelines. Finally, we also extend our approach to consider (vi) interactions between guideline recommendations and patient status. In such a way, we provide physicians with a framework for investigating different ways to reconcile guidelines, coping (with different modalities) with the focused interactions.

Keywords: Computer-interpretable clinical guidelines · Knowledge representation and ontologies · Comorbidities · Combining medical guidelines

1 Introduction

Clinical practice guidelines are the major tool that has been introduced to grant both the quality and the standardization of healthcare services, on the basis of evidence-based recommendations. The adoption of computerized approaches to acquire, represent, execute and reason with Computer–Interpretable Guidelines (CIGs henceforth) can provide crucial additional advantages. Therefore, in the last twenty years, many different approaches and projects have been developed to manage CIGs (consider, e.g., the book [1] and the recent survey [2]). One of such approaches is GLARE (Guideline Acquisition, Representation and Execution) [3].

By definition, clinical guidelines address specific clinical circumstances (i.e., specific pathologies). However, specific patients may be affected by more than one

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 95–108, 2015.
DOI: 10.1007/978-3-319-26585-8_7

pathology (*comorbid* patients). The treatment of such patients is one of the main challenges for the modern health care, also due to the aging of population, and the increase of chronic pathologies. The problem is that, unfortunately, in comorbid patients the treatments of single pathologies may interact with each other, and the approach of proposing an ad-hoc "combined" treatment to cope with each possible comorbidity does not scale up: "*Developing Clinical Practice Guidelines that explicitly address all potential comorbid diseases is not only difficult, but also impractical, and there is a need for formal methods that would allow combining several disease-specific clinical practice guidelines in order to customize them to a patient*" [4]. Thus, new methodologies are required to study the interactions between treatments, and to combine treatments: "*This sets up the urgent need of developing ways of merging multiple single-disease interventions to provide professionals' assistance to comorbid patients*" [5]. In the last years, several computer-based approaches have started to face this problem, aiming at providing physicians with different forms of support for managing multiple CIGs and their interactions (see Sect. 6).

In our previous work in this area, we have started to extend GLARE to cope with comorbid patients (notably, however, the methodologies we have proposed and we are going to propose in this paper are largely system-independent). Instead of aiming to provide a *fully-automatic* tool to merge two (or more) CIGs, we want to develop a suite of tools and methodologies to *support* physicians, providing them information and hints that may be helpful in their activity when facing multiple CIGs, thus following the *mixed initiative* paradigm proposed in artificial intelligence and human-computer interaction [6]. Our final goal is to provide a mixed-initiative approach to support physicians to (1) detect and analyze CIG interactions, (2) manage them and (3) merge (parts of) CIGs. Until now, we have focused on the first step only.

In this paper, we extend our approach to support CIG "*conciliation*", intended as a preliminary step to achieve interaction management and CIG merge. In Sect. 3, we identify different "*management options*" adopted by physicians when facing interactions. In Sect. 4, we provide suitable "reasoning" tools to support such options. For instance, a limited form of *goal-based planning* may be required to avoid an interaction, in case no suitable alternatives for the conflicting actions are present in the CIGs, while *temporal reasoning* can be useful both to avoid an undesired interaction and to enforce a desired one. Finally, in Sect. 5 we also extend our approach to consider interactions between guidelines recommendations and patient status. Our approach provides physicians with indications about how to achieve the chosen "*management option*" of the selected interaction (which has been previously detected and analyzed). Such indications are the basis to support physicians in their choice about how to manage interactions and merge CIGs, which is the long-term final goal of our work.

2 GLARE and Comorbidities: Previous Work

The GLARE (Guideline Acquisition, Representation and Execution) system [3] has been built starting from 1997 in a long-term cooperation between the Department of Computer Science of the Università del Piemonte Orientale in Alessandria and the ASU San Giovanni Battista in Turin (one of the largest hospitals in Italy). In GLARE a

CIG is represented by a hierarchical graph, whose nodes represent actions and whose arcs model the control relations between them. GLARE distinguishes between *atomic* actions (simple steps in a CIG) and *composite* actions (plans), which are defined in terms of their components. GLARE adopts five basic types of atomic actions: (i) *work actions*, i.e. actions that describe "external" actions that must be executed at given points of the CIG, (ii) *pharmaceutical actions*, specifying a drug (or drug category) to be administered and its dosage, (iii) *decision actions*, modelling the choice between different alternatives, (iv) *query actions*, i.e. requests of information (typically of patient's parameters), (v) *conclusions*, modelling the output of decision actions.

Starting from 2013, we are extending GLARE to cope with comorbidities. Up to now, we have focused only on interaction detection and analysis, developing an ontology of interactions and support tools based on it. The main results we obtained are described in [7–9]. However, to make this paper self-contained, two central issues are briefly reported here: our ontology of interactions and the facilities we provide for interaction detection and analysis.

Ontology of Interactions. We have devised an ontological model for clinical actions and interactions (see Fig. 1 and [7–9]) integrated as much as possible with the existing "consensus" medical ontologies, such as SNOMED CT [10] for clinical terms and ATC [11] for drug classification. We represent composite, work and pharmacological actions (light blue elements in the figure) at two levels: (i) the prototype of the action (concept *Action*), which is characterized by the effects (arc *hasEffect*) and possibly the prescribed drugs (arc *substance*), and (ii) the action in a specific CIG (concept *CIGaction*), which inherits all the properties of the prototype, but represents also the intentions of such action (arc *aimsTo*). Such distinction is needed because the same action in different CIGs can have different goals. Both effects and intentions are represented as variations (green elements) of the patient status (concept *Variation*), modelled as changes (arc *hasModality*, concept *Modality*) of an attribute of the patient status (arc *focusOn*,

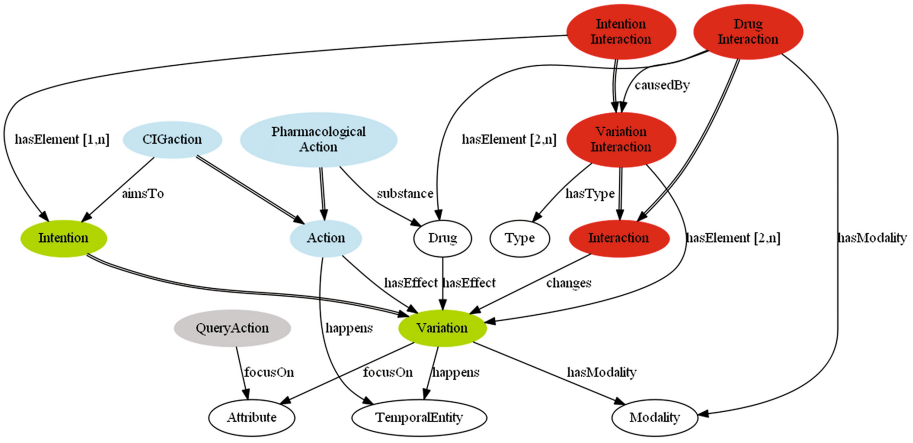


Fig. 1. Part of our ontology of action intentions, effects, drugs and interactions. Double arcs represent IS-A relationships.

concept *Attribute*). Modalities belong to the set {"*Increase*", "*Decrease*", "*Stability*"} (not shown in the figure due to space limitations) and further refinements of its elements (e.g., "*Stop*" can be a refinement of "*Decrease*") and attributes are modelled using SNOMED CT concepts. For instance, the variation "*Decrease Blood Pressure*" is modelled by the variation of the attribute "*Blood Pressure*" with modality "*Decreasing*". Variations are organized along an IS-A and PART-OF hierarchy.

The property *substance* of pharmaceutical actions describes the drug (concept *Drug*) they prescribe. Drugs are hierarchically organized (we use the ATC taxonomy) and each element of the hierarchy is described by the effects of its administration.

An interaction (red elements), in general, is described by the two elements it involves and by the variations it changes (arc *changes*).

Furthermore, an interaction between two or more variations (concept *VariationInteraction*) is represented by its type (arc *hasType*), whose value belongs to the set {"*Concordance"*, *"Discordance"*, "*Independence"*} and further refinements (e.g., "*Opposite"*). An *IntentionInteraction* is a particular case of *VariationInteraction* that involves at least one intention.

Interactions between drugs (concept *DrugInteraction*) are related to the modality of the variation (arc *hasModality*) that the interaction causes in the variation identified by the arc *changes*. Often, an interaction between two drugs is caused by an interaction between two of their effects. To model such information, the property *causedBy* relates a drug interaction to a variation interaction.

In [9] we have extended such a model to cope with time, by characterizing actions, variations and intentions through a temporal attribute (arc *happens*, concept *TemporalEntity*), which represents the execution time (for the actions), the time in which the variation should happen (for variations) and the time when the physician expects the intention will be accomplished (for intentions). Temporal knowledge is represented through (qualitative and quantitative) temporal constraints between temporal entities.

Interaction Detection and Analysis Support. Our approach supports automatic detection of the possible interactions between each pair of *focused* actions in two (or more) CIGs. If the focuses contain all the CIGs actions, an all-to-all analysis is performed. However, given the dimension of "real" CIGs, too many interactions would be returned. Often, when studying comorbidities, physicians focus on specific subproblems. For instance, given a patient, a physician may focus on the interactions between the next CIG actions to be executed. In such cases, an all-to-all approach would not be useful to them. For such a reason, in [8] we have proposed a mixed-initiative approach allowing the physician to navigate CIGs at different levels of abstraction (possibly following the *top-down refinement* methodology), allowing her/him to *focus* on *relevant* (to the specific case/problem under analysis) parts of the CIGs in order to look for possible interactions. In [9] we have extended the detection to support the *temporal* detection of interactions. Indeed, a non-temporal analysis can only detect *hypothetically possible* interactions between actions in different CIGs, identifying, e.g., a potential conflict between their goals (or effects). However, as long as no temporal analysis is performed, such an interaction is only "*potential*": *actual* interactions occur *in time*, i.e., just in the case that the considered goals or effects *overlap in time*.

3 Interaction Management Options

On the basis of the experience of physicians cooperating with GLARE, and considering the medical literature (see e.g., [12, 13]), we have identified a set of "*management options*" to manage interactions. Notably, such options are not mutually exclusive: indeed, in several practical cases, many options are possible, and the physicians have to choose between them. It is worth stressing that our goal is to produce limited and controlled changes in the treatments that respect, as much as possible, the constraints suggested by the original CIGs. This is due to the fact that clinical guidelines (from which CIGs derive) are "best practice" care plans developed following medical *evidence*, and any change to them results in a non-evidence-based recommendation, which must be exploited carefully and confirmed by the physician's opinion before being applied. The ontological knowledge in Sect. 2 can be exploited by physicians to discriminate between "desired" and "undesired" interactions (knowledge-based reasoning tools to support them in such a decision, like the ones proposed in [14], are out of the scope of this paper; see future work in the concluding section). Even if our approach can be applied on an arbitrary number of CIGs, in the rest of the paper we will consider an analysis of the interactions between two CIGs. The generalization on three or more CIGs is trivial.

In the medical literature, undesired interactions are usually *avoided* or are managed through *adjustments* of the CIGs. The avoidance of an interaction is required, e.g., when the interaction imperils the patient's life or when it compromises some of the intentions of an action, or when it causes undesired side effects. Two main options are provided: the *safe alternative* option and the *temporal avoidance* option.

Safe Alternative: the *safe alternative* option consists in the choice of alternative CIG paths in which the considered interaction does not occur. For instance, Telaprevir is an antiviral drug used in the treatment for hepatitis C virus, while Midazolam is an anxiolytic drug used as premedication before endoscopy. The concurrent administration with Telaprevir increases the effects of Midazolam and can lead, for instance, to breathing difficulties. For such reasons the interaction between them should be avoided. An alternative for Telaprevir is the treatment with Ribavirin, or with Temazepam or Lorazepam instead of Midazolam.

Temporal Avoidance: interactions can also be *temporally avoided*. To do so, interacting actions can be executed at times such that the interaction cannot actually occur (i.e., the two hypothetically interacting variations do not overlap in time – see also [9]). Some examples of temporal avoidance are the administration of drugs at different times of the day (to avoid side effects) and/or the provisional suspension of a CIG because of the incompatibility with another one. For instance, Calcium Carbonate administration causes a temporary (4–5 h) alkalinization of the urine, which interacts with the Nalidixic Acid absorption. Since the Calcium Carbonate, used for the treatment of gastroesophageal reflux, is taken after the meals, the physician can estimate a time for the Nalidixic Acid administration that avoids the interaction.

Not all the undesired interactions need to be avoided. In some cases, CIGs can be *adjusted* to manage the cases in which the interactions arise. We support three main

management options to this purpose: the *dosage adjustment* (for drug interactions), the *effect monitoring* and the *interaction mitigation*.

Dosage Adjustment: the *adjustment of the dosage* is particularly useful for those drug interactions that cause a deviation (increase/decrease) in the intention of one of the two interacting actions (see Sect. 2). E.g., when Midazolam is intravenously administered, the above interaction with Telaprevir can be managed by halving the Midazolam dosage.

Effect Monitoring: in some cases, *monitoring the effects* of the interaction is enough (instead of, e.g., mitigating the effects through dosage adjustment). In particular, if an interaction causes a change of some parameters of the patient, they have to be monitored and evaluated by the physician during the span of time in which the interaction occurs. For instance, the interaction between Warfarin (an anticoagulant) and Erythromycin (an antibiotic) is often avoided due to the increase of the Warfarin effects, which can cause bleeding. However, in some cases, avoidance is not possible. Then, the Prothrombin time has to be monitored (to derive the International Normalized Ratio) to estimate the risk of bleeding. Obviously, if a serious risk is detected, other management options can be applied, e.g., the therapy can be suspended or the Warfarin dosage can be adjusted.

Interaction Mitigation: some interactions can cause undesired side effects. In such cases, a "shared" action that mitigates such effects can be added to the interacting CIGs. For instance, Telaprevir decreases the contraceptive effect of the Ethinyl estradiol. To mitigate such an interaction, the physician may suggest the additional use of two non-hormonal types of contraception.

On the other hand, *desired* interactions are those that, by *physician preference* or "by *CIG definition"*, should happen. A physician may desire an interaction, for instance, because the effects of the interacting actions are enhanced by the interaction. On the other side, an interaction is desired "by CIG definition" when two (or more) actions in the different CIGs can be replaced by a "common" action achieving their intentions. We provide two types of alignment to manage such cases: the *alignment* based on the *interaction* and the *alignment* based on the *intention*.

Interaction Alignment: to guarantee the occurrence of an interaction two requirements are needed: (i) the interacting actions must be executed and (ii) their execution times have to be such that their effects overlap in time.

For instance, when a patient suffering from edema has to be treated also for hypertension, the physician may decide to combine the Diuretics (administered for treating edema) with the ACE inhibitors (administered for treating hypertension), obtaining an enhancement of the anti-hypertensive effects.

Intention Alignment: in the case of intention alignment, the physician may want to "merge" two actions in a single one, executing it in a time that respects all the intentions of both the original actions. This alignment is useful, for instance, with duplicated actions (e.g., Aspirin administration, recommended as antithrombotic and as antipyretic) and with actions that can be replaced by a third one, achieving both their intentions (e.g., blood cholesterol measurement and creatinine measurement can be replaced by a single blood examination).

4 A Mixed-Initiative Methodology for CIG Conciliation

In this section, we explain how our system provides support to the management options previously described, by exploiting different Artificial Intelligence reasoning techniques to automatically annotate the CIGs with additional information/constraints about how to achieve the management option examined by the user. In particular, we propose three basic reasoning methodologies, namely a *backward CIG navigation* (NAV), *temporal reasoning* (TR) and a limited form of *goal-based planning* (GBP). Such methodologies are combined (as summarized in Table 1 in the following) in different ways to provide physicians with a user-friendly support to determine how the different management options can be obtained. The result is shown to the user-physician in the form of graphical annotations of the input CIGs, explaining how to relate the two CIGs to achieve the desired management. Notably, the above methodologies must be used in different ways, depending on whether the management option is being considered (i) while studying "abstractly" how to manage a possible interaction between two CIGs or (ii) while considering such an interaction in the context of the execution of the two CIGs on a specific patient. In the following, for the sake of brevity, we only focus on case (i).

Table 1. Reasoning techniques adopted to support management options.

Safe alternative	Temporal avoidance	Dose adj.	Effect monitoring	Interact mitigation	Interact align	Intention align
NAV or GBP	TR	other	GBP + TR	GBP + TR	NAV + TR	GBP + TR

Backward CIG navigation (NAV) is used to move back in the two CIGs, retrieving alternative CIG paths. In GLARE, as well as in most CIG formalisms, CIGs are represented by hierarchical graphs, and alternative paths originate from decision actions. Thus, we provide NAV through a search in a hierarchical graph, looking for alternative paths, originating from previous decision actions in the CIG. In particular, GLARE distinguishes between diagnostic and therapeutic decisions. While the former depend on the status of the patient, the latter totally depend on the physician's judgment. Thus, we constrain our approach to search for alternatives stemming from therapeutic decision nodes only. Standard graph searching algorithms (not discussed here for the sake of brevity) are used for NAV.

Goal-based planning (GBP). Given one (or more) intention, our goal-based planning returns a set of plan skeletons (i.e. sets of actions, without control arcs) to achieve such intention. Our algorithm is highly mixed-initiative, meaning that the user-physician drives the planning process, while the system performs the ontology search. We distinguish between two situations: a basic case (i), in which only a non-decomposable (through the PART-OF relation) intention *int* has to be accomplished, and a complex case (ii), in which more than one intention has to be accomplished.

For case (i), we need to retrieve from the ontology all the actions (*actset*) that satisfy (*hasEffect* arc) *int*. The algorithm consists of two phases. First, the set *actset* containing all the actions that have as effect (*hasEffect* arc, followed backward) *int* is retrieved[1]. Notice that the set *actset* also includes composite actions (i.e., actions whose components have been specified through PART-OF relations). Then, the physician may refine *actset* pruning away all the actions that s/he considers not relevant for the specific problem under examination (e.g., actions not applicable to the specific patient). At the end, *actset* contains all the promising (following the physician opinion) actions that satisfy *int*.

Case (ii) involves the management of more than one intention. It requires some modifications to the basic algorithm. When, for instance, two intentions int_1 and int_2 have to be satisfied, the procedure of action finding in the ontology (*basic_af* algorithm) is applied separately to the two intentions. Two resulting sets of actions are obtained, named $actset_1$ and $actset_2$. The system then returns the Cartesian product of $actset_1$ and $actset_2$ as possible solutions.

```
algorithm basic_af:
input: Intention int;
output: Set<Action>;
Set<Action> actset ← inverseof hasEffect1(int);
for each action act ∈ actset
  issuit ← the system asks the user if act is suitable
  if issuit == 'no' actset ← actset \ {act}
return actset;
```

Temporal Reasoning (TR). We ground our approach on the temporal reasoning facilities described in [9]. Basically, our STP-based temporal reasoning algorithms solve Simple Temporal Problems considering the temporal constraints extracted from part of the CIGs and from our ontology (and, possibly, from the log of the executed actions). Among the others, [9] provides facilities to hypothesize temporal constraints between CIGs (useful, e.g., to relate the execution times of some actions in the two CIGs), and, in such a context, to analyze whether an interaction must/can occur in time (and when, INT facility), to identify the possible execution time of future actions *"to avoid an interaction"* (TFA-AI) or *"to obtain an interaction"* (TFA-OI).

All the management options described in Sect. 3 (except dosage adjustment) can be obtained on the basis of the above reasoning techniques, as summarized in Table 1.

The **safe alternative** option can be achieved in two ways. (1) NAV is used to find the first (preceding) therapeutic decision action in each CIG originating an alternative path not causing the interaction. The decision actions are then annotated with the suggestion to follow alternative paths. (2) GBP can be used to add non-interacting alternatives to CIG actions, searching plans that satisfy the intentions of one of the interacting actions.

[1] Only those actions that are directly connected to *int* are returned, while their sub categories do not. Such a choice is adopted in order to limit the dimension of the output, making it manageable by user physicians.

Temporal avoidance can be obtained directly by applying the TFA-AI facility.

In **effect monitoring**, a variation of GBP above (not described here for space constraint) is used in order to find out a monitoring action (*query* action in GLARE) for the patient status attribute changed by the given interaction. Such an action is added (as a "shared action") to the two CIGs, together with the constraint stating that it must be executed during the interaction time. Such a time is determined through the INT temporal reasoning facility.

In **interaction mitigation**, GBP is used to determine a plan that satisfies an intention opposite (or discordant) with respect to the variation caused by the interaction. Then INT and TFA-OI are used in order to determine the temporal constraints such that the effects of the plan occur during the interaction.

In **interaction alignment**, NAV is used to retrieve all the paths that lead to the interaction and to suggest the physician to follow one of them, respecting the temporal constraints identified through TFA-OI.

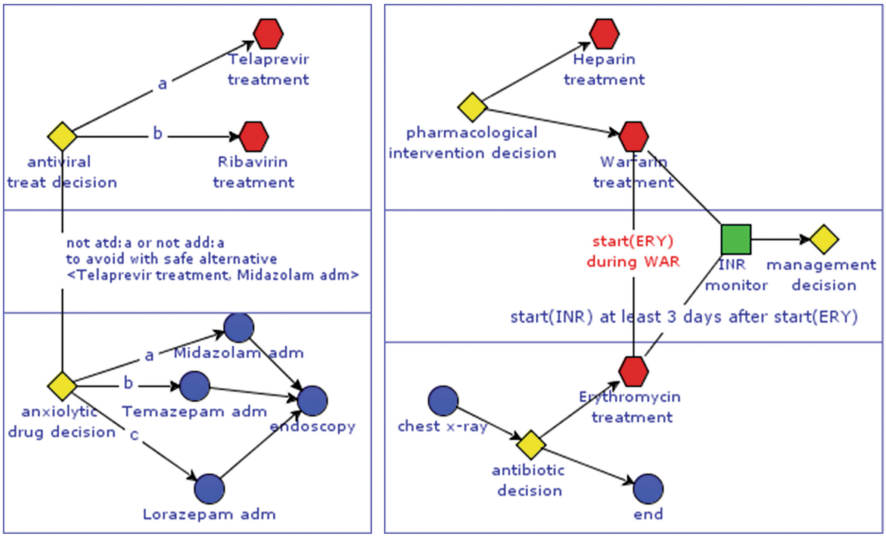
In **intention alignment**, GBP is used to find out a plan that achieves all the intentions of the replaced actions. Then temporal reasoning is adopted to find out the temporal constraints (on the new plan) needed to enforce that the new plan achieves its intentions at the time required by the input CIGs.

Finally, the **dosage adjustment** is easier, since it does not require any of the previous reasoning techniques. In fact, the system suggests an adjustment of the drug dosage opposite with respect to the variation caused by the interaction (i.e., an increase of the dosage in case of decreased variation and vice versa).

Table 1 summarizes how the different techniques are composed to achieve the different modalities.

In Fig. 2 we propose two examples of how our system annotates the input CIGs with the suggestions. Each example in the figure contains three boxes: the first and the third contain the CIG views, the middle one contains the added suggestions.

Figure 2 (a) shows a safe alternative management between the Telaprevir treatment (part of the CIG for hepatitis C virus, above) and Midazolam (premedication for endoscopy CIG, below). It shows the results of applying the safe alternative option to manage such an interaction. The system navigates backward the CIGs, finding "antiviral treatment decision" (for the first CIG) and "anxiolytic drug decision" for the second, and appends the suggestion to avoid a simultaneous decision in the two paths containing the interacting actions ("not atd:a or not add:a", where atd and add are abbreviations of the considered decisions). Figure 2 (b) shows the effect monitoring management between Warfarin treatment (in the CIG for venous thromboembolism) and Erythromycin (CIG for chest infection). The added action "INR monitoring" is retrieved by the system because it focuses on the attribute "Blood coagulation status" (which is an explicit SNOMED CT concept), which is the attribute decreased by the above interaction. Then, temporal reasoning is exploited to state that the interaction surely arises (at least) three days after the start of the treatment with the antibiotic (under the assumption, given by the user, that the Erythromycin treatment started during the Warfarin treatment). Then, a management decision is placed after the INR monitoring to decide, on the basis of the monitoring action, whether to continue the treatment, to suspend it or to manage the interaction with another option.



(a) **(b)**

Fig. 2. Graphical representation of the suggestions appended to the CIGs through the management options. (a) represents a *safe alternative* management, while (b) an *effect monitoring*.

5 Copying with Patient's Status

Until now, we focused on interactions between CIG actions only. However, in healthcare processes, also the patient's status is central and it influences the decision-making process. In this section, we expand our approach to cope with this aspect. In particular, following the schema already adopted for action interactions, we present (i) an expansion of our ontological model to represent the patient's status, (ii) a methodology to detect and analyse interactions regarding the patient's status and (iii) a modification of the above "management options" to cope with the new type of interactions.

Representing Patient's Status. To model the fact that an attribute of the patient's status (e.g., "*Blood Pressure*") has a particular value, we use the concept *Valorisation* (see Fig. 3). A valorisation is described by an *Attribute* (connected to *Valorisation* by the arc *valorises*), a value (concept *Value*, with possible values {*High*, *Low*, *Normal*} and further specifications, connected to *Valorisation* by the arc *hasValue*) and a valid time, modelled through a *TemporalEntity* (arc *hasValidity*), representing the time (time point or time interval) when the valorisation holds. We consider two sources of valorisations: the *entry points* of the CIG and the *query actions*. Entry points are the points in which the patient can start the CIG execution; query actions have been introduced in the previous sections. In general, a CIG can have one or more entry points and each of them is characterized by a description (defined by the guideline) of patient's status, which is valid by default for all the CIG duration (the *eligibility conditions* of the CIG).

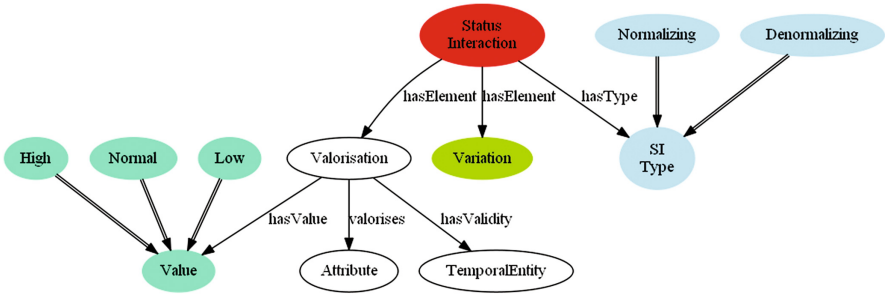


Fig. 3. Ontological representation of attribute valorisations and status interactions.

For instance, the entry point of a CIG for hypertension treatment characterizes the attribute "*Blood Pressure*" with value "*High*".

From the implementation point of view, we store patient data in a temporal database. Even if determinate intervals (i.e., with exactly known ending points) are possible, in most medical cases the valid time of data regarding the patient's status is *now-related* (e.g., *John has high blood pressure from May 10th to now*, asserted at time now = May 30th) [15]. We thus use the framework we devised to cope with now-related data [15]. In such a framework, the valid time of now-related data is represented by a triple: the starting time (May 10th in the example), the assertion time (May 30th) and the ending time ($+\infty$ in the example, but a definite bound can be represented). This notation represents the underlying semantics of now-related data: they *certainly hold* from the starting time to the assertion time, and *possibly hold* until the ending time.

Status Interaction Model and Detection. Avariation (intention or effect) interacts with an attribute valorisation if it focuses on the same *Attribute* of the valorisation, or on a sub-concept of it. We call such interactions *status interactions* (concept *StatusInteraction*) and distinguish them into two sub-categories (arc *hasType*): *Normalizing* (i.e., the modality of the variation brings the attribute to assume a normal value) or *Denormalizing*. For instance, the intention "*Decrease Blood Pressure*" interacts with the attribute valorisation "*High Blood Pressure*" with type "*Normalizing*". However, like other interactions in our model, if the time dimension is not considered a status interaction is only hypothetical [9]. Thus, also this type of interaction requires a two-step detection. In the first step, the "hypothetical" interaction is found querying the ontological knowledge. Then temporal reasoning is performed to verify the overlap between the time of the variation and the time of the validity of the status. Considering now-related valorisations, the overlap can be *certain* (if the certain interval between the starting point and now overlaps with the variation), or *possible* (if also the possible future bound is considered).

Managing Status Interactions. Copying with status interactions requires minimal modifications to the above explained management options. The main difference is that attribute valorisations are out of the control of the system (i.e., managing options cannot avoid them or move them along the time). Thus, when considering options like

safe alternative, the only element that can be modified (searching for alternatives through NAV or GBP) is the action causing the interacting variation (i.e., "the interacting action"). **Temporal avoidance** requires the same modification: the only element that can be moved in time to avoid the interaction is the interacting action. **Effect monitoring**, **interaction mitigation** (where the mitigating action, in general, tends to normalize the attribute), **interaction alignment** and **dosage adjustment** do not require modifications, provided that, where a temporal overlap is required, the overlap between the attribute valorisation and the variation is considered. On the other hand, **intention alignment** cannot be applied for status interactions.

As an example, we take into account interaction mitigation. Let us consider a patient affected by peptic ulcer that develops gout. The entry points of the CIG for the peptic ulcer add the attribute valorisation "*High risk of Gastric Bleeding*" to the patient status. In the treatment of gout, nonsteroidal anti-inflammatory drugs (NSAIDs) should be prescribed. However, the effect "*Increase risk of Gastric Bleeding*" of NSAIDs interacts with the previous valorisation with type "*Denormalizing*". In this case, we suppose that the physician decides to apply the interaction mitigation option. The GBP process extracts, from the ontology, the actions "*Misoprostol Administration*" and "*Proton Pump Inhibitor treatment*", which have the effect "*Decrease risk of Gastric Bleeding*". Then, the suggestion to execute one of those two actions after the NSAIDs is added to the CIGs to mitigate the interaction.

6 Related Work and Conclusions

Comorbidity management is a hot topic in the Medical Informatics community. In recent years, several approaches have been devised to cope with the merging or concurrent execution of CIGs. Considering *interaction detection* only, the approach in [14] is the most similar to our one, providing a CIG-independent conceptual model for medical actions and reasoning forms operating on it. Moreover, in [14] general rules are proposed in order to identify different types of interactions on the basis of such a knowledge, as a basis for managing such interactions.

On the other hand, several other approaches require the explicit insertion (for each set of CIGs used to cope with a type of comorbidity) of the possible interactions between CIG actions, and focus on the *merging* of CIGs. The approach in [4] and [16] uses constraint logic programming to identify and address adverse interactions. A constraint logic programming (CLP) model is derived from the combination of logical models that represent each CIG and a mitigation algorithm is applied to detect and mitigate interactions. On the other hand, Sánchez-Garzón et al. [17] propose an agent-based approach to guideline merging. Each CIG is modeled by an agent with hierarchical planning capabilities. The result is obtained through the coordination of all the agents and respects the recommendations of each CIG. Riaño et al. represent CIGs as sets of clinical actions that are modelled into an ontology [18]. Treatments are first unified in a unique treatment and then a set of "combination rules" is applied to detect and avoid possible interactions. In [5] a model-based automatic merge of CIGs is proposed, through the definition of a combining operator. Jafarpour and Abidi [19] use

semantic-web rules and an ontology for the merging criteria. Given such knowledge, an Execution Engine merges CIGs according to merge criteria.

The main goal of the work in this paper is quite different. While the above approaches aim at providing (mostly) *automatic* tools to *merge* CIGs, mostly on the basis of a *single reasoning paradigm* (e.g., constraint logic programming or agent-based planning), here we only focus on proposing to physicians a framework for exploring different alternative ways to *reconcile* (specific parts of) guidelines. To achieve such a goal, we propose a *mixed-initiative* approach, in which multiple and *hybrid* forms of "reasoning" are adopted to *suggest* physicians how different conciliations can be achieved, depending on which "management option" they choose. A prototype implementing the entire approach is under development. However, we have already implemented part of the ontology and of the reasoning algorithms, testing their performances and the quality of the results. The ontological knowledge is implemented using OWL DL and SWRL rules (see [7]). Given its size, querying the ontology is the most onerous task. However, since most of the reasoning can be performed offline, online querying provides real time responses. In addition, performances can be improved by considering less detailed hierarchies of variations/intentions. Temporal reasoning has been implemented using STP constraint propagation (see [9]). Backward CIG navigation uses a breadth-first search algorithm. Given the number of elements, both tasks can be easily performed online. To evaluate the quality of the results, we applied our approach to well-known interactions, comparing our outputs with the suggestions in the literature. Not surprisingly, the quality of the results of our non-temporal analysis depends on the accuracy of the modeled knowledge. On the other hand, since (through STP reasoning) we enforce strict consistency between temporal constraints, our temporal suggestions tend to be stricter (more constrained) than the ones in the literature.

In the future, we aim at extending our approach with (i) reasoning capabilities which, operating on the ontology of actions and interactions, may provide physicians suggestions on which "management options" are the most appropriate to cope with the specific interaction at hand (see [14]), (ii) user-friendly facilities to navigate and compare different CIG conciliations in order to support physicians in the choice between them, and (iii) facilities to (locally) merge (part of) CIGs on the basis of the chosen conciliations. In such extensions, we will follow our philosophy: GLARE aims at providing physicians with "useful" knowledge (and recommendations) to *support* their decisions, but medical decisions are *deliberately not automatized*, since they pertain to physicians.

Acknowledgments. The work described in paper was partially supported by Compagnia di San Paolo, in the Ginseng project. The authors are also very indebted with Prof. Gianpaolo Molino for his continuous support and for many inspiring suggestions and comments.

References

1. Ten Teije, A., Miksch, S., Lucas, P. (eds.): Computer-Based Medical Guidelines and Protocols: A Primer and Current Trends. IOS Press, Amsterdam (2008)
2. Peleg, M.: Computer-interpretable clinical guidelines: A methodological review. J. Biomed. Inform. **46**, 744–763 (2013)

3. Terenziani, P., Molino, G., Torchio, M.: A modular approach for representing and executing clinical guidelines. Artif. Intell. Med. **23**, 249–276 (2001)
4. Michalowski, M., Wilk, S., Michalowski, W., Lin, D., Farion, K., Mohapatra, S.: Using constraint logic programming to implement iterative actions and numerical measures during mitigation of concurrently applied clinical practice guidelines. In: Peek, N., Marín Morales, R., Peleg, M. (eds.) AIME 2013. LNCS, vol. 7885, pp. 17–22. Springer, Heidelberg (2013)
5. Riaño, D., Collado, A.: Model-based combination of treatments for the management of chronic comorbid patients. In: Peek, N., Marín Morales, R., Peleg, M. (eds.) AIME 2013. LNCS, vol. 7885, pp. 11–16. Springer, Heidelberg (2013)
6. Horvitz, E.: Uncertainty, Action, and Interaction: In Pursuit of Mixed-Initiative Computing (1999)
7. Piovesan, L., Molino, G., Terenziani, P.: An ontological knowledge and multiple abstraction level decision support system in healthcare. Decis. Anal. **1**(8), 1–24 (2014)
8. Piovesan, L., Molino, G., Terenziani, P.: Supporting multi-level user-driven detection of guideline interactions. In: Proceedings of HEALTHINF, pp. 413–422. Scitepress (2015)
9. Piovesan, L., Anselma, L., Terenziani, P.: Temporal detection of guideline interactions. In: Proceedings of HEALTHINF, pp. 40–50. Scitepress (2015)
10. International Health Terminology Standards Development Organisation: SNOMED Clinical Terms. http://www.ihtsdo.org/snomed-ct
11. WHO Collaborating Centre for Drug Statistics Methodology: Anatomical Therapeutic Chemical classification system. http://www.whocc.no/atc/
12. Edwards, I.R., Aronson, J.K.: Adverse drug reactions: definitions, diagnosis, and management. The Lancet. **356**, 1255–1259 (2000)
13. Burger, D., Back, D., Buggisch, P., Buti, M., Craxí, A., Foster, G., Klinker, H., Larrey, D., Nikitin, I., Pol, S., Puoti, M., Romero-Gómez, M., Wedemeyer, H., Zeuzem, S.: Clinical management of drug-drug interactions in HCV therapy: challenges and solutions. J. Hepatol. **58**, 792–800 (2013)
14. Zamborlini, V., Hoekstra, R., da Silveira, M., Pruski, C., ten Teije, A., van Harmelen, F.: A conceptual model for detecting interactions among medical recommendations in clinical guidelines. In: Janowicz, K., Schlobach, S., Lambrix, P., Hyvönen, E. (eds.) EKAW 2014. LNCS, vol. 8876, pp. 591–606. Springer, Heidelberg (2014)
15. Anselma, L., Piovesan, L., Sattar, A., Stantic, B., Terenziani, P.: A general approach to represent and query now-relative medical data in relational databases. In: Holmes, J.H., Bellazzi, R., Sacchi, L., Peek, N. (eds.) AIME 2015. LNCS, vol. 9105, pp. 327–331. Springer, Heidelberg (2015)
16. Wilk, S., Michalowski, W., Michalowski, M., Farion, K., Hing, M.M., Mohapatra, S.: Mitigation of adverse interactions in pairs of clinical practice guidelines using constraint logic programming. J. Biomed. Inform. **46**, 341–353 (2013)
17. Sánchez-Garzón, I., Fdez-Olivares, J., Onaindía, E., Milla, G., Jordán, J., Castejón, P.: A multi-agent planning approach for the generation of personalized treatment plans of comorbid patients. In: Peek, N., Marín Morales, R., Peleg, M. (eds.) AIME 2013. LNCS, vol. 7885, pp. 23–27. Springer, Heidelberg (2013)
18. López-Vallverdú, J.A., Riaño, D., Collado, A.: Rule-based combination of comorbid treatments for chronic diseases applied to hypertension, diabetes mellitus and heart failure. In: Lenz, R., Miksch, S., Peleg, M., Reichert, M., Riaño, D., ten Teije, A. (eds.) ProHealth 2012 and KR4HC 2012. LNCS, vol. 7738, pp. 30–41. Springer, Heidelberg (2013)
19. Jafarpour, B., Abidi, S.S.R.: Merging disease-specific clinical guidelines to handle comorbidities in a clinical decision support setting. In: Peek, N., Marín Morales, R., Peleg, M. (eds.) AIME 2013. LNCS, vol. 7885, pp. 28–32. Springer, Heidelberg (2013)

Mobile Process and Decision Support

Implementation of a Distributed Guideline-Based Decision Support Model Within a Patient-Guidance Framework

Erez Shalom[1(✉)], Yuval Shahar[1], Ayelet Goldstein[1], Elior Ariel[1], Moshe Sheinberger[1], Nick Fung[2], Val Jones[2], and Boris van Schooten[2]

[1] The Medical Informatics Research Center, Department of Information System Engineering, Ben Gurion University of the Negev, Beersheba, Israel
{erezsh, yshahar, gayelet, eliorar, sheinmos}@bgu.ac.il
[2] University of Twente, Enschede, The Netherlands
{l.s.n.fung, V.M.Jones}@utwente.nl,
b.vanschooten@rrd.nl

Abstract. We report on new *projection engine* which was developed in order to implement a distributed guideline-based decision support system (DSS) within the European project MobiGuide.In this model, small portions of the guideline knowledge are projected, i.e. 'downloaded', from a central DSS server to a local DSS in the patient's mobile device, which then applies that knowledge using the mobile device's local resources. Furthermore, the projection engine generates guideline projections which are adapted to the patient's previously defined preferences and, implicitly, to the patient's current context, which is embodied in the projected knowledge. We evaluated this distributed guideline application model for two complex guidelines: one for Gestational Diabetes Mellitus, and one for Atrial Fibrillation. We found that the initial specification of what we refer to as the *customized* guideline should be in the terms of the distributed DSS, i.e., include two levels: one for the central DSS, and one for the local DSS. In addition, we found significant differences between the customized, distributed versions of the two guidelines, indicating further research directions and possibly additional ways to analyze and characterize guidelines.

Keywords: Clinical guidelines · Decision support · Distributed computing

1 Introduction

1.1 The Need for Distributed Decision Support

Clinical Guidelines (GLs) are a well-established method for enhancing the quality of care and for reducing costs [1]. Usually, the GL's recommendations are addressed solely to the care providers, such as the physicians, and not to the patients, and typically at the point of care, and not at home. Thus, existing frameworks for providing automated GL-based decision support focus mainly on supporting the care providers at the point of care. However, the role of the patient in the process of care is becoming

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 111–125, 2015.
DOI: 10.1007/978-3-319-26585-8_8

more and more central. In addition, many GL recommendations address patient behavior, especially in the case of chronic illnesses, where treatment must be continued outside the hospital and partly managed by the patient. Therefore we believe that patients, and in particular, chronic patients, should be empowered to manage their own disease by extending both the GLs and the GL-based decision-support frameworks to provide guidance for patients outside the standard clinical settings. An architecture for patient guidance could provide the patients themselves with appropriate GL-based alerts and recommendations, and could also monitor and react to changes in the patient's personal environment. Both objectives can be achieved through the use of applications running on mobile devices. The potential of mobile devices for assisting patients in the process of self-care has already been demonstrated; one compelling example relates to the goal of improving adherence to taking medications at home [2]. Ideally, recommendations should be personalized, in the sense of considering the patient's personal schedule, important external events, and personal preferences corresponding to changing contexts. Such personalization can be achieved through an extension of the GL, customizing it to consider explicitly non-clinical contexts that were not accounted for in the original GL, such as the patient living alone, or the battery status of the mobile device [3].

However, monitoring alone is insufficient; mobile-based applications also need to factor in patient education as well as up to date, real-time GL-based recommendations [4, 5]. In addition, the full GL, which might need to be frequently updated, and the patient's full medical history, might need to reside on a central server, which will have a complete view of the patient's course of disease, as well as of the relevant clinical knowledge. Connection to such a server cannot always be guaranteed; the server may become unavailable due to an unexpected overload or other technical factors. In addition, some GL-based tasks should be delegated to the mobile device as a matter of course to prevent overburdening the central server. This can be achieved by distributing commonly occurring, computationally intensive tasks, especially when based on high-frequency data, such as continuous monitoring and detection of cardiac arrhythmias in patients prone to such a disorder, close to the patient. An example of such an intense calculation is the detection of potential patterns of Atrial Fibrillation (AF) episodes in an individual patient, based on a set of high frequency ECG sensor signals. Thus, *there is a need to distribute the decision support process from the central server to a local mobile device*.

1.2 The Distributed Decision Support Model in the MobiGuide Project

To address the challenges associated with real-time patient guidance systems, the European Union's MobiGuide project [3] was initiated. The main goal of this project is to develop a distributed patient guidance system which integrates historical hospital records and current monitoring data into a Personal Health Record (PHR) accessible by patients and physicians, and providing personalized, secure, clinical-guideline-based guidance both inside and outside standard clinical environments. The distributed model of such a framework might be implemented as a service oriented architecture [6], which might be more suitable for distributing a process inside a hospital. However, in the case

of the MobiGuide project, we have chosen to split the architecture into two main components: a *back-end Decision Support System* (*BE-DSS*) residing on a server system (this could be a cloud server, or, as in our case, on-premise servers in hospitals), and a *mobile DSS* (*mDSS*) residing on the patient's mobile device. The local mDSS is necessary to distribute computationally intensive monitoring and decision-making processes, with respect to data and knowledge requirements, at the local device level.

Previously [7], we had presented very briefly a new model we had developed for a distributed DSS, which we have implemented within the MobiGuide framework: In this model, portions of the clinical *guideline* (*GL*) which can be identified as a self-contained executable knowledge packages to be potentially applied in the mDSS, are projected to the local mDSS. To the best of our knowledge, only the GLARE GL application framework [8] introduced explicitly the concept of distributed GL-based decision support, implemented by managing several agents that interact in different clinical settings, called "contexts". However, that extension was intended to deal mostly with human interaction and communication, and with human resources management; the agents were human; and none of the agents mentioned was the *patient*.

Figure 1 demonstrates the MobiGuide projection workflow model: After the physician initiates the application of the *Gestational Diabetes Mellitus* (*GDM*) guideline (number 1), the BE-DSS retrieves the full GDM GL from the GL *knowledge-base* (KB) server (number 2), but sends only the specific sub-plan "monitor blood glucose once a week", which was previously tagged as a *projected plan,* to the mDSS; this sub-plan (representing the current treatment plan for that individual, and personalized with patient preferences and current context) is then applied by the mobile device [9] (number 3). At any time, a certain predefined *breakout* temporal pattern might be detected by the mDSS (as part of the projected sub-plan) or by the BE-DSS. When a breakout pattern is detected by the mDSS, the mobile device sends a message to the

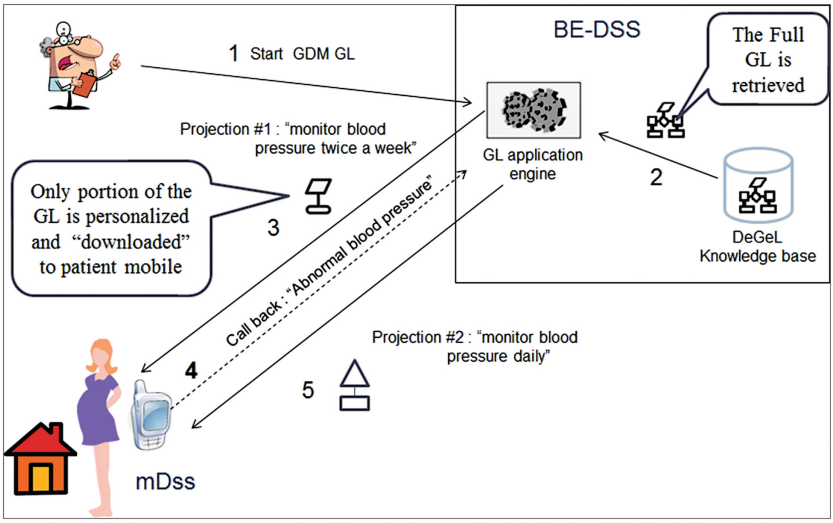


Fig. 1. An example of a projection model workflow.

BE-DSS to "take control" and continue the application of the GL, indicating which pattern caused the breakout. This message to the BE-DSS from the mDSS is called a *callback*, and is also predefined within the projection (number 4). A callback, or detection of such a pattern by the BE-DSS, will in some circumstances lead to the sending of a new projection file by the BE-DSS to the mDSS (number 5) (e.g. effecting a change in the treatment plan).

For example, the breakout pattern "abnormal blood-pressure" detected by the mDSS (or BE-DSS) might lead to stopping the "twice a week blood-pressure measurement" sub-plan, and starting a new "daily blood-pressure measurement" sub-plan. In both cases, the switch between the current sub-plan and the new sub-plan, occurs at the BE-DSS level, which sends a new projection file to the mDSS. A projection file includes one or more projected plans, which will be activated immediately and applied by the mobile device. Note that The BE-DSS can manage, when necessary, the patient in centralised fashion, but it does not use the projection and callback mechanisms for that purpose. The BE-DSS sends the mobile device's API a direct notification with the recommendation details, including a potential interaction with the patient. The notification bypasses the projection and callback mechanisms.

1.3 Aspects Affecting the Distribution of Decision-Support

The decision which actions or overall management plans should be distributed to a set of mobile devices, and which need to stay centralized, is not a trivial one, and is affected by several factors, as we have previously pointed out [7].

For example, consider the clinical task of detecting cardiac arrhythmias, such as AF. We can continuously monitor each patient's high-frequency ECG sensor signals by means of patient-worn biosensors linked via Bluetooth to the mobile device, and detect a pattern of AF that can be abstracted from these signals by the mDSS and sent to the central PHR to support a guideline-based recommendation to the patient and/or physician by the BE-DSS. Such a distribution of labor, in which the AF detection for each patient is performed locally by the mDSS is natural, and prevents an over burdening of the central BE-DSS server. Furthermore, the local mDSS is also essential for continuity of care when for some reason there is no internet connection to the central DSS; we would still want the patient to be provided with alerts relevant to the latest guideline by which she is being managed.

However, not all decisions can be made locally on the mobile (despite the ever increasing processing power and storage capacity of smartphones). Some decisions require the full historical (longitudinal) patient medical record, including all laboratory tests, physical examination, diagnoses, hospitalizations, procedures, and other interventions, which for practical reasons cannot, and for security and privacy reasons should not, reside on the mobile device. Furthermore, in some cases, the decision to be made is part of a long-term plan in the complete clinical guideline knowledge base (which is continuously maintained and updated as needed by medical domain experts), and in other cases, broader medical declarative knowledge (and interpretation) may be needed in order to detect a meaningful clinical pattern which may require a switch to another branch in the guideline, or even to a completely different guideline; such knowledge

resides only on the central knowledge-base server. In such cases, as described above, we wish the mDSS, when encountering certain local predefined (temporal) patterns, to send a callback message to the BE-DSS, asking for further instructions, resulting in a BE-DSS recommendation, and possibly even in a completely new projected guideline or guideline branch.

For these reasons we adopted a distribution policy based on a number of factors. The policy is applied starting at the stage of medical knowledge engineering (i.e., when customizing the GL for distributed care), in order to determine whether various decisions and actions should be applied at the BE-DSS level or at the mDSS level, following the general principles that we proposed in [7]. The factors are: the actor of the decision (patient or physician); the temporal horizon of future recommendations (e.g. immediate alerts by the mobile device when some value is out of range, versus longer-term guideline-based decisions made at the server); the data and knowledge resources needed for the decision; the need for PHR access, and a consideration of where a potential personalization of the guideline should reside. These principles need to be considered by the knowledge engineer and the expert physicians during the knowledge specification phase. However, once it is decided that a decision can be applied by the mDSS, i.e., can support the patient when he/she is not with the physician, the relevant guideline knowledge needs to be projected to the mDSS.

In the sequel we describe how the projection model was implemented in the context of MobiGuide project, and how it was evaluated in the case of two complex GLs for management of GDM and AF. In general, the implementation of the projection model includes two main tasks: specification of the GL in terms of a distributed DSS, and development of the projection engine as part of the GL application engine; one of the subtasks of that engine is generating and personalizing the projections. In the following sections we will describe how we implemented these two tasks.

2 Specifying the Guideline in the Terms of a Distributed DSS

2.1 Choosing Projected Plans in the GL

Specifying a GL for application by a fully distributed DSS requires a different strategy from traditional GL knowledge engineering. As explained above, it involves multiple new challenges, such as: deciding at which level (mDSS or BE-DSS) each plan or decision should be placed, deciding which action or plan needs to be performed, and deciding which breakout patterns should trigger callbacks to the BE-DSS. This process is performed during the GL specification phase by a knowledge engineer in collaboration with expert physicians, as part of the process of creating a consensus regarding the GL [10].

Based on our experience, we have outlined several principles for selecting which plans in the GL should be projected [7], some of which were mentioned in Sect. 1.3. These characteristics helped us understand when a certain decision task should be delegated to the local mDSS, and when it should best be left to the central BE-DSS. In general, computations that are intense, though not necessarily algorithmically challenging, and that can be easily distributed and performed using only local patient data,

such as AF detection given only a single patient's data, should be distributed to the multiple local devices. Computations that are algorithmically complex (e.g., require complex temporal pattern detection) and that require the patient's historical, longitudinal medical record, and/or the full knowledge base, should be performed centrally.

Table 1 shows several examples of the distribution of decision making between the BE-DSS and the mDSS for the GDM and AF GLs.

Figure 2 shows how we implemented the tagging of plans as plans to be projected as part of the GL, using the GESHER knowledge acquisition tool [11], as part of the *Digital electronic Guideline Library* (DeGeL) [12], in the case of the GDM GL: At specification time, the knowledge engineer checked the "*is-projected*" property of the sub-plans that were determined as sub-plans that need to run at the mDSS level, in this case the "monitor Ketonuria daily" sub-plan (number 1), and the "monitor [for Ketonuria] twice a week" sub-plan (number 2). Note that in both cases, two sub-plans are tagged as *projected*: the first is a periodic sub-plan for measuring the ketonuria each day (the circular arrow shape); the second is a monitoring sub-plan (the hexagonal shape), which in fact monitors for a breakout pattern (in this case, the pattern "two positive values in a week of ketonuria"), and which, if detected, asks the patient a question regarding their diet and

Table 1. Examples for choosing decisions at the different DSS levels

Domain	Level of decision	Decision description	Explanation
AF	BE-DSS	Is the patient eligible for the "Pill-in–the-pocket" emergency plan?	The relevant data are stored in the PHR and are therefore not accessible by the mobile device
GDM	mDSS	"when a pattern of two weeks of normal blood-glucose is detected, send a callback to BE-DSS"	Requires only a simple calculation based on relatively short-term accumulating daily blood-glucose values
AF	mDSS	Monitor AF episodes	Abstracted from high-frequency ECG signals generated by a local sensor whose data are accessible to the mobile; requires intensive computation using local data, which is best performed locally for each individual patient
GDM	BE-DSS	Does the Ketonuria abstraction have a negative value over the past week AND the Diet has not been changed since the last visit?	The data about visits to clinicians resides in the PHR, accessible to the back-end DSS, and do not exist locally in the mobile device
GDM and AF	BE-DSS	Context change	Context change is affecting multiple GL plans and requires a global view of the complete guideline

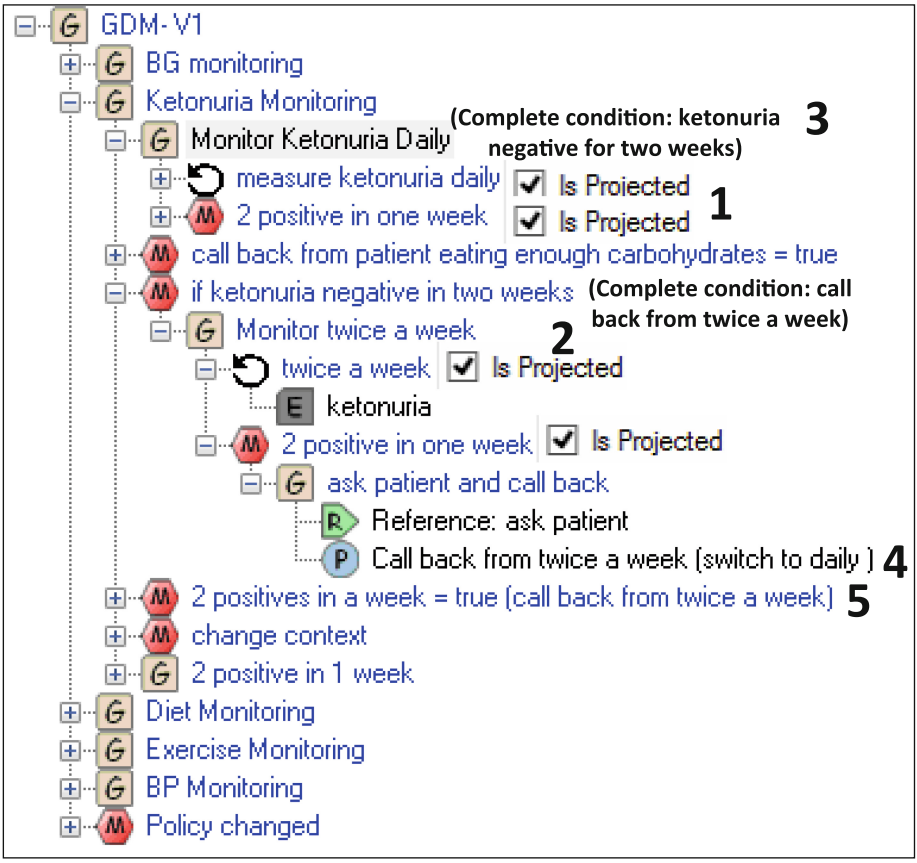


Fig. 2. The GESHER interface and the tagging of projected plans in the GL, in the case of the GDM GL

triggers a callback to the BE-DSS to determine how to proceed. Note that several projected plans (depending on their internal eligibility criteria and the GL's overall workflow) might be sent in the same projection file to the mobile device. The basic language used for representing the GL, underlying the GESHER GL-specification tool, is Asbru [13], and its hybrid-Asbru extensions [12]; we have augmented it using the projection and callback tags.

2.2 Definition of Appropriate Callbacks

As explained above, projections and callbacks support a continuous dialog between the BE-DSS and the mDSS. Thus, the GL created to support a distributed DSS is also specified in terms of messages between the mDSS and the BE-DSS. This includes specifying the projections to send the mobile device, the breakout patterns to be detected by the mDSS, and the associated callbacks from the mDSS to the BE-DSS.

On the other hand, at the BE-DSS level, monitoring plans are "listening" to all of the relevant breakout patterns, and to callback messages coming from the mDSS, which might cause the BE-DSS to send the mDSS a message to stop an existing projected plan, or to send a new projected plan to be activated.

The implementation of this unique dialog in terms of GL specification is shown also in Fig. 2: first, "monitor ketonuria daily" is projected to the mobile (number 1), and accordingly a monitoring sub-plan to detect the temporal pattern "ketonuria has been negative for two weeks" is activated, in this case, at the BE-DSS level, as determined by the knowledge engineer who built the parallel workflow for this section of the GL (note that in this particular case, it could also be applied by the mDSS by projecting it to the mobile device). When that sub-plan is triggered (by another part of the BE-DSS, an *intelligent monitoring* module that monitors the data for knowledge-based temporal patterns and that is subscribed to the urine measurements reaching the PHR server (see Sect. 3, number 2), two events occur: (1) The *complete condition* (in the terms of the Asbru language [13], in which MobiGuide GLs are specified) for the daily monitoring sub-plan is triggered, thus causing the BE-DSS to send a projection to the mDSS to stop it (number 3), and (2) a new sub-plan, to reduce the frequency of monitoring to twice a week is started and is projected by the BE-DSS to the mDSS.

Note that the new projected "monitor ketonuria twice a week" sub-plan, which replaces the originally projected "measure ketonuria daily" sub-plan, includes a call-back instruction (number 4) to the BE-DSS, in case the mDSS detects the breakout pattern of two positive values of ketonuria in a week. When the mDSS detects this breakout pattern, a callback is sent to the BE-DSS. This call-back is constantly monitored (through the intelligent monitoring module) by a specific monitoring sub-plan (number 5); thus, when the callback arrives at the BE-DSS, it causes the BE-DSS to send a stop message regarding the sub-plan for twice weekly monitoring (number 2), and to project to the mDSS a daily monitoring plan.

3 The Projection Engine

In order to support the distributed DSS projection model, we developed a new component, the *projection engine*, which extends the functionality of our existing GL application engine [14]. The extended BE-DSS architecture is shown in Fig. 3: the GL application engine gets the GL knowledge from the GL KB, and applies it. When the GL application engine finishes, the projection engine examines which parts of the existing activated sub-plans need to be projected. The projection engine then retrieves the preferences and personalized contexts [3] of the patients from the data integrator through the data and knowledge services layer, and may also perform queries via the intelligent monitoring module (e.g., to get the current context of the patient). Then, the projection engine generates the projection file, which is sent by the GL application engine to the mDSS through the *Body Area Network* (BAN) back-end server [15], which mediates between the BE-DSS and the mDSS.

The projection engine produce two types of projections: (1) Declarative projections including concepts (simple abstractions), and personal (patient-specific) events that induce predefined customized contexts in the GL, within which the guideline's actions

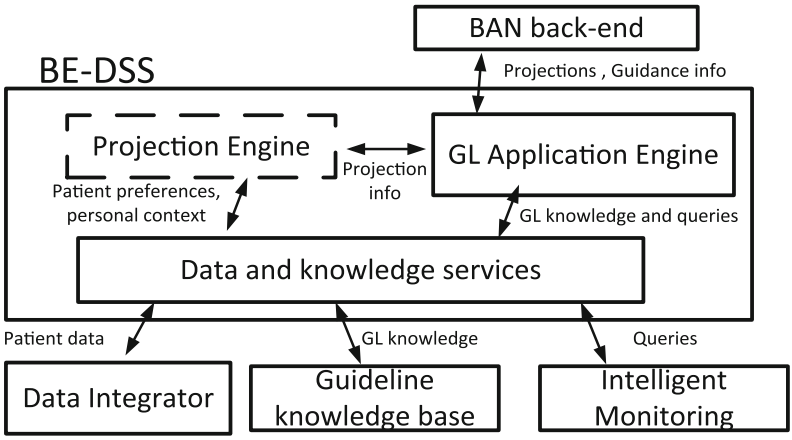


Fig. 3. High level architecture of the BE-DSS and the projection engine

might be modified, and (2) Procedural projections – including sub-plans for general treatments or for specific treatments relating to personalized-contexts.

3.1 Declarative Projections

One of the innovative features of the MobiGuide system is support for personalization. Personalization of the GL occurs when the patient is enrolled as a user of the MobiGuide system. In this step, predefined clinical contexts, already part of the customized GL (e.g., a High Carbohydrate Meal context, or an Irregular Schedule context), are retrieved from the knowledge base, and are shown to the patient so that he or she can choose the corresponding personal events that induce these predefined contexts (e.g. "Wedding" or "Vacation"). The mapping between the personal events and the contexts is stored in the PHR and is sent to the mDSS before starting the GL application session, as part of the declarative projection. The mDSS uses the declarative projection to send to the patient's interface the list of personal events that were selected during the initial enrollment session as inducing certain predefined contexts in the customized guideline. For example, for a given patient, "Vacation" or "Holiday" events might induce the predefined context "irregular schedule", and a wedding event might induce the predefined context "High Carbohydrate Meal". For example, a patient might register on her smartphone the personal event "at work", the mDSS reports this event to the PHR. The event, for her, induces (at the level of the BE-DSS) the "Regular-Schedule" context. If this patient subsequently reports registers the personal event "holiday," this will be then reported by the mDSS to the PHR, inducing, for her, (at the level of the BE-DSS) the context "Irregular Schedule", thereby leading the BE-DSS to send the mobile device a second projection to be applied by the mDSS, with a different periodic monitoring rate. The BE-DSS always generates the projections to be sent to the mobile device according to the current (possibly induced by a personal event) context of the patient.

The declarative projection also includes the *Quality of Data* (QoD) [16] information used by the *Quality of Data Broker* located at the mobile, in order, for example, to "ignore" invalid input data (e.g. an out of range blood pressure value).

3.2 Procedural Projections

Procedural projections are generated at run-time by the projection engine. The engine checks the "*is projected*" property for each sub-plan (see Fig. 2), and if it is set to "true", the sub-plan's corresponding projection file is generated and added to the projection collection (see below for more details about projection collection). Otherwise, the BE-DSS engine continues to apply the sub-plan. Also, as part of the projection process, the specific thresholds of the patients (e.g. personal target exercises levels), and the preferences relating to the personal contexts of the patient are retrieved from the PHR and are set in the projections.

Each projection file is decomposed into several "unit-projections". Each unit-projection is a single sub-plan. For example, the sub-plan "monitoring blood glucose once a week" is decomposed into two unit-projections: (1) the sub-plan for blood glucose measurement schedule, and (2) the sub-plan to monitor several days of high fasting blood glucose levels, signifying that the patient is not well controlled. Each projection and each unit within a projection starts independently, as soon as it arrives at the mobile device. Each unit has its own set of internal temporal constraints, including temporal relations among different actions. Thus, all important temporal-constraint knowledge resides (is encapsulated) within the projected units and not among them, hence concurrency problems are not an issue.

Before building a projection on the fly, the projection engine checks what is the current context of the patient, (e.g. "Regular Schedule", which is induced by the "at work" event); it then retrieves all of the patient's scheduling preferences for this context (e.g., days and hours of reminders), and modifies the projections accordingly. For example, in unit-projection "20102", the time to activate reminders to the patient is set to "8:00" which is the preferred hour by the patient to get reminders in the context "Semi-routine Schedule" (Fig. 4).

In addition, the projection file contains two lists of IDs: one for the sub-plans to be stopped, and one for the sub-plans to be started. When the BE-DSS is triggered (e.g. by an incoming callback from mDSS, or through detection of a breakout pattern), all affected sub-plans which are needed to transit into their complete state are aggregated by the projection engine into a unified *stop-list*. On the other hand, sub-plans that need to start are aggregated into a *start-list*. Thus, the mDSS is always "up-to-date" with respect to the sub-plans to be stopped, or to be started at the local level (the rest of the plans currently applied by the mDSS are assumed by default to be continuing).

If the patient's mobile device crashes, the projection engine recovers the last procedural projections sent to the mobile device, and resends them to the mDSS. To support this functionality, we added to the BE-DSS a new *projected-plans* collection to store the different projections generated during the GL application session by the projection engine. Table 2 shows this collection in the context of the projection shown in Fig. 2. Each row represents a unit-projection in the collection which has a link to the

```
projection("19857", id="184");

stop("20091,20092");

start("20102,20130");


unitProjection("20102","Semi-Routine Daily BG Fasting measurement") {

        while (true) {
                waitPeriodic("1,2,3,4,5,6,7", "8:00", null);
                event = createEvent();
                event.patientDataEntry("4985","BG Fasting","numeric","1 hour");
                event.insert();
        }
}


unitProjection("20130","2 abnormal measurements in past week") {

        annotateTemporal("or",new String[] {
                        "event.getNumber(4985)>=150",
                        "event.getNumber(4986)>=150",
                        "event.getNumber(4987)>=150",
                        "event.getNumber(4988)>=150"
                }, "abnormal_BG", "date" );

        while (true) {
                waitTemporalQuery("count >= 2", "abnormal_BG", "8 calendardays");
                callback("5112", "2 abnormal values in BG were found in your
                        measurements in the past week,
                        system is calculating another schedule for you for daily BG measurement");
        }
}
```

Fig. 4. An example of a projection file sent to the mDSS from the BE-DSS, containing two unit-projections

projection ID it belongs to, a unit-projection ID, the timestamp showing when it was sent to the mDSS, and status (started or stopped). Note that all unit-projections shown in Table 2 are sent from the same projection at the same time. The mDSS uses these properties to manage the execution of the sub-plans running locally (for example, it might stop the "daily blood pressure monitoring" sub-plan #22 from the stop-list, and start a new plan for "twice a week" monitoring sub-plan #23 from the start-list).

3.3 Implementation of Personalization with Dynamic Projections

At projection time, except for patient preferences for days and hours of reminders, the projection engine replaces all pre-defined knowledge thresholds with real values. An example of a knowledge threshold is the personal target level for physical exercise. Values above this threshold are abnormal. As the threshold values might be changed from time to time, writing their explicit value in the projection file will be hard to maintain. Instead, the threshold knowledge ID is written as a variable name (a string)

Table 2. The *projected-plans* data structure, which stores the different unit-projections generated by the projection engine

Projection ID	Unit-projection ID	SentDate	Status
184	20091	10/5/14/14:00:00:00	stop
184	20092	10/5/14/14:00:00:00	stop
184	20102	10/5/14/14:00:00:00	stop
184	20130	10/5/14/14:00:00:00	stop

```
unitProjection("20010","Weekly METS") {

        while (true) {
                waitPeriodic("7", "7:00", null);
                if (temporalQuery("sum >= <$5066$>", "5065", "7 days")) {
                        patientNotification("5162","Enhorabuena, el ejercicio
                                                ayuda al buen control. Sigue así.");
                } else {
                        patientNotification("5163","Recuerda que hacer ejercicio
                                                es importante para tu bienestar
                                                y para mantener un buen control
                                                de la glucosa.");
                }
        }
}
```

Fig. 5. The knowledge thresholds in the case of calculating the threshhold for weekly exercise. Patient notification texts are shown in Spanish (for the GDM pilot in Spain)

circumscribed by triangular brackets; at projection time, the variable is replaced by the real value from the knowledge base. Figure 5 shows an example for unit-projection "20010": at design-time, the threshold knowledge ID is set, for example, to the variable " <5066> ", (which in this case denotes the exercise target level). At projection time, the projection engine replaces this string with the real value; in this case, all thresholds are set to "5".

Another dynamic projection behaviour is handling drug prescriptions: as drug prescriptions are patient-specific they cannot be part of the GL knowledge. Thus, the projection engine adds each valid drug medication (plus its appropriate dose and schedule) it finds in the patient's personal record dynamically as a unit-projection, and converts the dates to start and stop the drug to total days. Each drug is then personalized according to the current context of the patient, for example, in the case that the time for taking a drug is set to "after lunch", this time is generated according to the lunch hour belongs to the current context so that the mDSS receives the specific hour for taking drug. In addition, the projection engine add a drug to the stop-list it is not valid anymore.

3.4 Evaluation of the Projection Model

Together with expert-physicians we specified the GDM [17] and AF GLs using the GESHER knowledge acquisition tool [11]. A complete discussion of the knowledge acquisition process is out of the scope of this paper. Each GL took approximately 3 months to specify in detail, in collaboration with domain experts.

Following that phase, we identified projected plans in the GL. Most of the projected plans were periodic and monitoring sub-plans. Projected periodic sub-plans are plans in which some action should be performed periodically by the mDSS (see Sect. 2). Table 3 shows the distribution of the projected plans between the GLs, and their characteristics. Altogether, we tagged 39 projected plans to the GDM GL, and 20 for the AF GL. Note that in the case of the AF GL, most of the projections were of periodic

Table 3. The distribution of the projections projected plans **between the GLs**

	Projected Periodic sub-plans	Projected Monitoring sub-plans	Callbacks	Decisions made at BE-DSS	Decisions made at mDSS
GDM	22	17	16	44	34
AF	18	2	2	24	36

sub-plans and not of monitoring sub-plans, and there are only 2 callbacks. That is because most of this GL's actions can be handled by the mobile device, which thus very rarely needs the BE-DSS to change the projection. In contrast, in the case of the GDM GL, more decisions are made at the BE-DSS level, since more decisions in that GL require additional data (such as past and future visits) and care-giver confirmations, both of which are accessibly only to the BE-DSS.

4 Summary and Discussion

We have presented an innovative framework for the distributed application of clinical guidelines through a two-tiered architecture, which integrates a central DSS server with multiple local mobile devices that monitor individual patients, and thus splits the computational tasks of applying a GL between them. The projection and callback mechanism that we have implemented support a continuous dialog between the BE-DSS and the mDSS, splitting the decision-support tasks between the central BE-DSS, which is linked to the overall medical knowledge base and to the patient's EMR, and the local mDSS, and exploit the relative advantages of the different computational architectures and their respective access to clinical data and medical knowledge.

To the best of our knowledge, the distributed GL application architecture that we have designed and implemented is entirely new; even previous studies suggesting the distribution of GL application, mostly referred to the assignment of different tasks to different [human] agents, per their specialty [8]. We believe that the principles underlying this two-tiered architecture are rather general, and support both functional (e.g., send recommendations to patient) and non-functional (e.g., efficiency, security) requirements.

We have implemented both the distributed GL specification and the distributed GL application in the case of the GDM and AF guidelines. We found that these two GLs create very different projection and callback profiles when represented as distributed GLs. The difference might represent a profound difference between the characteristics of the two GLs, possibly due to the increased need of accessing the EMR for the patient's history, in the case of the GDM GL. Representing additional GLs in a distributed format might lead to a better understanding of GL characteristics and to additional insights regarding the differences amongst them.

We are in the final year of the four year MobiGuide project. and are currently running a pilot study to test the feasibility of using a distributed DSS architecture to manage patients using the two guidelines: The GDM guideline, applied in collaboration

with the Sabadell Hospital in Barcelona, Spain, and the AF guideline, applied in collaboration with the Fondazione Salvatore Maugeri, Pavia, Italy.

Acknowledgements. The MobiGuide project (http://www.mobiguide-project.eu/) has received funding from the EU's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 287811.

References

1. Quaglini, S., Ciccarese, P., Micieli, G., et al.: Guideline application for decision making in ischemic stroke (GLADIS) study group. Stud. Health Technol. Inform. **101**, 75–87 (2004)
2. Granger, B.B., Bosworth, H.B.: Medication adherence: emerging use of technology. CurrOpinCardiol. **26**(4), 279–287 (2011)
3. Peleg, M., Shahar, Y., Quaglini, S.: Making healthcare more accessible, better, faster, and cheaper: the MobiGuide Project. Eur. J. ePractice: Issue Mob. eHealth **20**, 5–20 (2014)
4. Chomutare, T., Fernandez-Luque, L., Arsand, E., Hartvigsen, G.: Features of mobile diabetes applications: review of the literature and analysis of current applications compared against evidence-based guidelines. J. Med. Internet Res. **13**(3), e65 (2011). doi:10.2196/jmir.1874
5. Farmer, A.J., Gibson, O.J., Dudley, C., Bryden, K., Hayton, P.M., Tarassenko, L., Neil, A.: A randomized controlled trial of the effect of real-time telemedicine support on glycemic control in young adults with type 1 diabetes. Diabetes Care **28**(11), 2697–2702 (2005)
6. Besana, Paolo, Barker, Adam: Towards decentralised clinical decision support systems. In: Brahnam, Sheryl, Jain, Lakhmi C. (eds.) Adv. Comput. Intell. Paradigms in Healthcare 5. SCI, vol. 326, pp. 27–44. Springer, Heidelberg (2010)
7. Shalom, E., Shahar, Y., Goldstein, A., et al.: Enhancing Guideline-based decision support with distributed computation through local mobile application. In: Prohealth 2014
8. Bottrighi, G.M., Montani, S., Terenziani, P., Torchio, M.: Supporting a distributed execution of clinical guidelines. Comput. Methods Programs Biomed. **112**, 200–210 (2013)
9. Sacchi, L., Fux, A., Napolitano, C., Panzarasaa, S., Peleg, M., Quaglini, S., Shalom, E., Soffer, P., Tormene, P.: Patient-tailored Workflow Patterns from Clinical Practice Guidelines Recommendations. In: Medinfo 2013 (2013)
10. Shalom, E., Shahar, Y., Taieb-Maimon, M., Lunenfeld, E., Bar, G., Yarkoni, A., Young, O., Martins, S.B., Vaszar, L.T., Goldstein, M.K., Liel, Y., Leibowitz, A., Marom, T., Lunenfeld, E.: A quantitative evaluation of a methodology for collaborative specification of clinical guidelines at multiple representation levels. J. BioMed. Inf. **41**(6), 889–903 (2008)
11. Hatsek, A., Shahar, Y., Taieb-Maimon, M., Shalom, E., Klimov, D., Lunenfeld, E.: A scalable architecture for incremental specification and maintenance of procedural and declarative clinical decision-support knowledge. Open Med. Info. J. **4**, 255–277 (2010)
12. Shahar, Y., Young, O., Shalom, E., Galperin, M., Mayaffit, A., Moskovitch, R., et al.: A framework for a distributed, hybrid, multiple-ontology clinical-guideline library, and automated guideline-support tools. J. Biomed. Inform. **37**(5), 325–344 (2004)
13. Shahar, Y., Miksch, S., Johnson, P.: The Asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines. A I. Med. **14**, 29–51 (1998)
14. Shalom, E., Shahar, Y., Parmet, Y., Lunenfeld, E.: A multiple-scenario assessment of the effect of a continuous-care, guideline-based decision support system on clinicians' compliance to clinical guidelines. Int. J. Med. Inform. (in press). doi:10.1016/j.ijmedinf.2015.01.004

15. Jones, V., Bults, R., Konstantas, D., Vierhout, P.: Healthcare PANs: Personal Area Networks for trauma care and home care. In: Fourth International Symposium on Wireless Personal Multimedia Communications (WPMC), Aalborg, Denmark (2001)
16. Larburu, N., Van-Schooten, B., Shalom, E., Fung, N., Hermens, H., Jones, V.A: Quality Aware Mobile Decision Support System for Patients with Chronic Illnesses. In: Prohealth 2015 (2015)
17. García-Sáez, G., Rigla, M., Martínez-Sarriegui, I., Shalom, E., Peleg, M., Broens, T., Pons, B., Caballero-Ruíz, E., Gómez, E.J., Hernando, M,E.: Patient-oriented computerized clinical guidelines for mobile decision support in gestational diabetes. J. Diabetes Sci. Technol. (2014). doi:10.1177/1932296814526492

A Quality-of-Data Aware Mobile Decision Support System for Patients with Chronic Illnesses

Nekane Larburu[1(✉)], Boris van Schooten[1,2], Erez Shalom[3], Nick Fung[1], Marten van Sinderen[1], Hermie Hermens[1,2], and Val Jones[1(✉)]

[1] University of Twente, PO Box 217, Enschede, The Netherlands
{n.larbururubio, l.s.n.fung, m.j.vansinderen, v.m.jones}@utwente.nl, {b.vanschooten, h.hermens}@rrd.nl
[2] Roessingh Research and Development, Enschede, The Netherlands
[3] The Medical Informatics Research Center, Ben Gurion University of the Negev, Beer-Sheva, Israel
erezsh@bgu.ac.il

Abstract. We present a mobile decision support system (mDSS) which runs on a patient Body Area Network consisting of a smartphone and a set of biosensors. Quality-of-Data (QoD) awareness in decision making is achieved by means of a component known as the Quality-of-Data Broker, which also runs on the smartphone. The QoD-aware mDSS collaborates with a more sophisticated decision support system running on a fixed back-end server in order to provide distributed decision support. This distributed decision support system has been implemented as part of a larger system developed during the European project MobiGuide. The MobiGuide system is a guideline-based Patient Guidance System designed to assist patients in the management of chronic illnesses. The system, including the QOD-aware mDSS, has been validated by clinicians and is being evaluated in patient pilots against two clinical guidelines.

Keywords: Decision support · Computer-interpretable clinical guidelines · Knowledge representation for healthcare processes · Context-aware healthcare processes · Mobile process and task support in healthcare

1 Introduction

We present the design and implementation of a quality-aware mobile decision support system (mDSS) [1]. The mDSS forms part of a larger system developed during the IST MobiGuide project, in which a guideline-based Patient Guidance System (PGS) designed to assist patients in the management of chronic illnesses is researched, developed and evaluated. The MobiGuide PGS supports the patient and their medical team in adhering to best evidence as encapsulated in clinical practice guidelines. Moreover it supports communication between them, information sharing and shared decision making between patient and clinician. The goal is to support mobile, guideline-based monitoring and management, supporting independence whilst preserving safety.

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 126–139, 2015.
DOI: 10.1007/978-3-319-26585-8_9

The mDSS is part of a distributed Decision Support System (DSS). The knowledge-base of the distributed DSS is based on the knowledge encapsulated in Clinical Guidelines. The mobile part is implemented, along with other components, on a smartphone, as part of a Body Area Network (BAN) which performs patient monitoring by means of body-worn or mobile sensors and delivers guideline-based recommendations to patients via a smartphone interface.

We define a BAN as a body worn network of communicating devices, incorporating a processing platform (e.g. smartphone). In the case of a health BAN, the devices may include medical devices such as biosensors as well as general purpose devices (e.g. alarm buttons). BAN data such as measurements from biosensors may be processed locally on the BAN or sent to a remote system for processing, or a combination of the two. In 2001 we proposed the first application of BAN technology in healthcare [2] to support trauma care and home care. A number of health BANs for patient monitoring were prototyped and trialled during the IST Mobihealth project. In subsequent research health BAN applications were developed for a range of chronic conditions and BAN applications were augmented with context awareness [3]. Real time support for clinical guidelines was proposed in [4] and adaptive feedback, augmenting telemonitoring with teletreatment, was added [5].

In MobiGuide we extend mobile health research by distributing decision support functionality between the patient's mobile system and a fixed back-end system; a feature shared with ubiquitous healthcare systems such as [6, 7]. However, in MobiGuide we also incorporate clinical decision support based on clinical guideline knowledge and introduce quality of data awareness into the formalized clinical guidelines which form the knowledge bases of the distributed decision support systems.

Quality-of-Data (QoD) awareness is based on augmentation of clinical guidelines with quality information during knowledge engineering and by labelling data with quality labels at run time so that decision making can be informed by quality of clinical data. Technological context and the associated impact on quality of clinical data are handled by the Quality-of-Data Broker (QoD Broker), which runs on the BAN. The QoD aware mDSS can run standalone on the BAN if necessary but normally collaborates with the more advanced DSS system running on the back-end.

In MobiGuide we focus on two patient groups: patients with Atrial Fibrillation (AF) and pregnant women with Gestational Diabetes Mellitus (GDM). The knowledge bases of the AF and GDM applications are based respectively on clinical guidelines [8, 9]. The MobiGuide system is designed to be generic, hence any well formulated clinical guideline could be used as a basis for a MobiGuide application for another clinical condition, assuming the appropriate knowledge engineering effort to derive a Computer Interpretable Guideline (CIG) from the narrative guideline.

This paper describes the mobile decision support system (mDSS), how QoD awareness is achieved via the QoD Broker, and how the mDSS collaborates with the back-end decision support system (BE DSS) to provide distributed decision support. Section 2 describes the knowledge engineering phase; specifically how guideline knowledge is transformed into a knowledge base and how quality information is

introduced at this stage. Section 3 presents the model for distribution of decision support between the BE DSS and the mDSS. Section 4 describes the QoD-aware mDSS and its relation to the back-end system. Sections 5 and 6 describe respectively the QoD Broker and the mDSS. Discussion and conclusions are found in Sect. 7.

2 Formalizing Clinical Guidelines

Clinical guidelines bring together the best and latest scientifically proven knowledge about how to manage and treat a particular condition and as such represent current medical consensus. They are developed by panels of top medical experts who review evidence from clinical trials and scientific literature in order to support evidence-based care. Most guidelines are written in natural language, however, and in order to integrate a guideline into an automated DSS the narrative guideline must be formalized to produce a Computer Interpretable Guideline (CIG). In the formalization step the guideline is analysed and carefully transformed into a semantically equivalent computer interpretable version expressed in a formal language such as Asbru [10].

Based on the knowledge acquisition methodologies of [11, 12], the guideline is first adapted to local practices and the tacit knowledge elicited from the narrative text, resulting in a local narrative consensus which is then marked up with semantic labels. This labelled, semi-structured text is then converted into a semi-formal representation which, in the MobiGuide project, takes the form of "parallel workflows" representing the sequence of tasks leading to clinical recommendations [13]. These workflows are then transformed into an executable form. As part of the analysis of the guideline, the narrative guideline and parallel workflows are also converted into a process model to identify possible options for distributing the required decision support functionality across the distributed DSS [14]. In this model, guideline knowledge is represented as a network of data flow processes, each of which encapsulates a separable portion of the guideline knowledge and represents, by definition, a unit operation that can be executed in parallel with the others. In this way, the model facilitates the identification of concurrent and similar tasks for distribution and allows a detailed exploration of the different possible distribution options.

In MobiGuide formalization is followed by two other steps during knowledge engineering: customization and personalization. These steps enable integration of context information and personalization of guidelines in order to improve effectiveness / efficacy of disease management whilst preserving patient safety by adding context awareness to the guideline and adapting it to the individual patient. The customisation step extends the CIG, for all patients, with different possible contexts; the personalisation step instantiates the customised CIG for an individual patient. During customization the CIG is extended with the possible contexts that could affect patient guidance. These contexts include personal context information, such as whether the patient has support at home or how their daily routine may change for example in holiday contexts or at social events such as weddings. As part of this step technological context information, expressed in terms of quality of data (QoD), is also added to the CIG.

Our definition of 'technological context' [15], which is aligned with Dey's [16], is the technological information, often expressed in terms of Quality of Service (QoS), provided by a collection of technological resources which characterize the treatment of a patient. The performance variations of technological resources (e.g. motion artefacts, battery level or poor internet connectivity) that characterize technological context affect the QoS of the system. As a result, the quality of the output of technological resources (i.e. the quality of the clinical data) will also be affected (see Sect. 5). Therefore, we augment the CIG with technological context information expressed in terms of QoD. This augmentation of the knowledge is performed by medical practitioners in collaboration with requirements engineers. First, requirements engineers prepare for each clinical variable (in each treatment) a "QoD effect table" that contains the five QoD dimensions that we adopted for our research (see Sect. 5). Medical practitioners determine via semi structured interviews the potential impact of each QoD dimension on treatment and how the treatment should be adapted to enhance patient safety. Requirements engineers include this information in the "QoD effect table". When the medical practitioners have validated the "QoD effect table" it is merged into the treatment scenarios and represented as data flow diagrams. Accordingly, the data flow diagrams cover the impact of QoD on different treatments. In case potential inconsistencies or conflicting conditions are encountered, medical practitioners modify the diagrams. Once the medical practitioners have validated the data flow diagrams, the information is incorporated into the formalized guideline. Subsequently these treatment adaptation mechanisms are validated with a live application of the telemedicine system that runs the executable QoD-aware CIG in the DSS. The resulting "customized" CIG defines how treatment is to be adapted for all patients according to the different possible contexts. Points where individual patient preferences can be taken into account are also specified in the customization step [17].

Personalization of the CIG takes place during a patient-physician encounter when they define together the concepts, specific to this individual patient, that will induce the contexts defined in the previous customization step and specify patient preferences (such as preferred timing of measurements). The resulting augmented CIG reflects the real state of the patient and allows him/her to receive decision-support suited to the context, based on the system's knowledge base which contains recommendations approved by physicians. These patient preferences can then be taken into account during CIG execution, enabling personalized recommendations to be delivered at appropriate times.

As a result, in the knowledge engineering phase the knowledge-base of the DSS (the augmented CIG) is extended amongst others with recommendations adapted to variations in QoD. In the operational phase incoming data (e.g. patient data from sensors) is annotated with quality labels by the QoD Broker. Together these two enable the DSS to be QoD-aware, so that the safety of the patient can be enhanced even when technological disruptions occur.

At the end of this process, the resulting (augmented) CIG is a formalized, customized, personalized and QoD aware version of the guideline. The two augmented CIGs for AF and GDM in MobiGuide are documented in [17].

3 Distributed Decision Support

In order to provide decision support to the patient anytime anywhere, the MobiGuide system incorporates, amongst other components, two decision support systems: one on the patient's smartphone (the mDSS) and one on the back-end server (BE-DSS). Although the BE-DSS, based on a continuous guideline application engine [18], has more resources available than a smartphone to perform complex data processing, it is dependent on a reliable mobile communications infrastructure for receiving patient data acquired by the BAN; this may not always be available. Therefore, by distributing some functionality to the patient's smartphone, the resources of which may be too limited for some complex decision support, the MobiGuide distributed DSS supports real-time operation independent of the network environment with the mobile part providing data input, basic processing, feedback, and guidance even if the network is temporarily unavailable. Furthermore, delegating processing to patients' mobile systems supports scalability of the service to large patient populations by processing raw bulk data locally and providing only the necessary summaries to the BE-DSS. For example, heart rate and physical activity level data are processed entirely locally on the BAN.

The mechanism for distributing knowledge and processing responsibilities between the mDSS and the BE DSS is known as Projection. In order to determine how to delegate parts of the decision support to the mDSS, several factors are considered:

- The actor of the decision (patient or physician), since it is more appropriate for the mDSS to provide decision support to patients only;
- The temporal horizon of future recommendations, whether they are alerts, for example, which require immediate patient attention and should therefore be performed by the mDSS, or longer-term decisions which are less dependent on reliable connectivity and can, as a result, be performed by the BE-DSS despite potential intermittent loss of connectivity;
- The data and knowledge resources needed for the decision compared to the resources available on the smartphone;
- The need for data stored in the PHR (Personal Health Record), which may not be accessible outside the hospital due to security and privacy considerations;
- The dependencies between different parts of the decision support, which can be identified, for example, by modelling the guideline as a network of concurrent processes (Sect. 2); and
- A consideration of where a potential personalization of the guideline should reside.

These principles need to be considered by the knowledge engineer and expert physicians during the knowledge specification phase. Once these factors are decided, the BE-DSS delegates procedural knowledge to the mDSS by creating and sending procedural directives called projections which incorporate the delegated procedural knowledge and the contextual information from the PHR (e.g. patient preferences and clinical history) that are required to interpret the raw BAN data. A procedural projection is a simplified decision procedure that can run stand-alone on the mDSS to handle decisions of part of the GL, typically for time spans ranging from days to months. It may eventually be replaced by another projection if the mDSS signals

exceptional circumstances and/or the BE-DSS decides to change the procedure. Projections contain mostly fixed schedules (such as measurement or medicine schedules), along with the conditions which could trigger schedule changes, such as QoD related conditions (see Fig. 2). The mDSS executes the schedules stand-alone until one of the change triggers occur. Then it sends a signal, a callback, to the BE-DSS which triggers the BE-DSS to change the projections if necessary. By a judicious choice of procedural knowledge to project to the mDSS, callbacks will occur at relatively low frequency, thus reducing the risks and effects associated with loss of connectivity to the back-end. While optimal functioning of the system does require the network to be available at regular intervals, the systems designed to degrade gracefully if the network is unavailable for longer periods of time.

Projections are subdivided into unit projections, which can run as parallel processes. The BE-DSS can send multiple new unit projections plus a directive to stop previously running projections in a single message. The main directives comprising a projection are detailed in Sect. 6, but typically, a projection contains a declarative part, which tags items in the mDSS database according to certain criteria, and a procedural part, which is usually a wait loop which triggers on a particular event or time. Figure 1 shows an example projection which represents a condition in the GDM guideline (two abnormal blood glucose measurements within one week) which triggers a recommendation to change the blood glucose measurement schedule. Although not explicitly shown in Fig. 1, projections may also include details of the clinical effects of quality of data. Data with insufficient quality may, for example, be tagged differently by being given a different ID and may, as a result, trigger different procedures.

```
unitProjection("20095","2 abnormal measurements in past week") {
annotateTemporal("or", [
"event.getNumber(4985)>=150",
"event.getNumber(4986)>=150",
"event.getNumber(4987)>=150",
"event.getNumber(4988)>=150"
], "abnormal_BG", "date");
while (true) {
waitTemporalQuery("count >= 2", "abnormal_BG", "8 calendardays");
callback("5111", "2 abnormal values in BG were found in your measurements in
the past week, system is calculating another schedule for you"); } }
```

Fig. 1. Example projection. The annotateTemporal statement defines the condition under which a record or set of records is annotated with a particular tag. In this case, it tags a set of events as abnormal_BG if one or more blood glucose (BG) measurements over 150 occur in one calendar day. The number 4985-4988 represent BG measurements at particular times of the day with quality higher than “very low”. The wait loop at the bottom waits for at least two abnormal_BGs to occur within 8 calendar days, then sends a callback.

4 The Quality-of-Data Aware mDSS

The focus of the paper is on the mobile part of the MobiGuide system, specifically the mDSS and its interaction with QoD Broker. This section focuses on the mDSS and the influence of QoD on the decisions output by the mDSS.

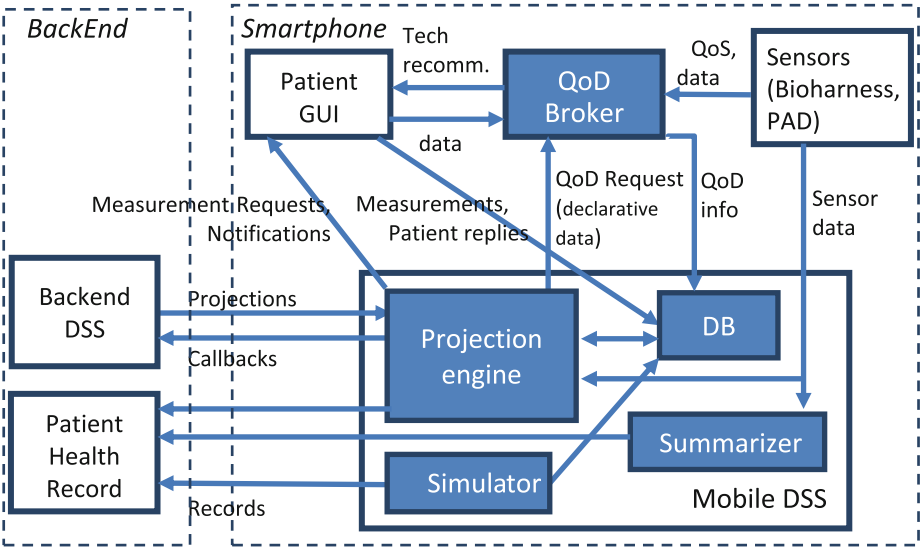


Fig. 2. Simplified architecture of the mobile QoD aware mDSS

Figure 2 shows the components comprising the mDSS and their main interaction with other components on the smartphone and the back-end. Communication proceeds through a piece of middleware called the BAN service, which handles authentication, network communication, and sensor handling. Furthermore, in case of temporary absence of connectivity, the BAN service can temporarily store all messages that are to be exchanged between the back-end and the smartphone in a queue. The mDSS components are:

- The projection engine, a scripting engine which takes care of starting, interpreting, and stopping projections. It receives projections from the BE-DSS and outputs callbacks for the BE-DSS, measurement requests to Patient GUI, and records for storage in the databases.
- The local database which stores records relevant for the mDSS functioning emanating from the projection engine, GUI or QoD Broker.
- The simulator, which enables a sequence of events to be simulated system-wide. It emits events which are then stored in the databases, namely the local database as well as the personal health record, and reacted to by the mDSS and BE-DSS.

- The summarizer, which produces summaries of AF monitoring sessions which are stored in the personal health record. Clinical abstractions include average heart rate and AF episode information.

5 The Quality-of-Data Broker

The Quality-of-Data (QoD) Broker is the component responsible for translating technological context information into QoD. Our design approach, aligned with that of Berti et al. [19], began with determination of the QoD dimensions relevant to the clinical application. The second step was to incorporate algorithms for quantifying QoD and finally we determined together with clinical specialists which is the relative QoD, since QoD requirements may diverge depending on the context (e.g. the specifics of treatment and patient condition).

QoD plays a major role in healthcare [20, 21]. The dimensions used to quantify QoD, sometimes termed QoD attributes, are highly diverse. They do not refer only to 'accuracy' or 'correctness' [22–25], but also to other quality attributes relevant for the user, aligned with 'fitness for use' theory [26, 27]. Based on literature and requirements of the clinical application, we express QoD according to five dimensions [15]:

- Accuracy: degree of correctness at which the attentive phenomenon is represented by the data. For example, if the heart rate (HR) sensor is not properly placed and the data is noisy, the accuracy of the monitored HR will be 'poor'.
- Dependability: degree of certainty that data is available (or complete), and can be used for meaningful decisions regardless of speed or accuracy. An example of 'poor' dependability is when it is not possible to measure HR due to the sensor unavailability due to lack of battery or when data connectivity is poor and data cannot be transmitted to the point of decision.
- Timeliness: time interval to transport data from source to destination. For example, HR data may contain a 'significant' delay for making a treatment decision on time due to data processing or transmission delay. This may lead to 'poor' timeliness and increase treatment risk if the patient needs to be notified immediately.
- Cost: amount of money required to obtain data for the decision-making process. Cost is a quality dimension that is addressed in very few QoD literature studies [28, 29], but is an important QoD dimension since it may affect other QoD dimensions, such as timeliness [29]. Ballou et al. [28] studied the tradeoff between cost and other QoD dimensions and found that 'in a majority of the cases the best solution in terms of error rate is the worst in terms of cost'. Moreover, medical practitioners attest the significance of cost in telemedicine systems, since it may influence treatment guidance. For example, if the patient pays more for roaming data than with Wi-Fi, when Wi-Fi is not available extra cost is needed, leading to 'poor' cost. Besides, if the roaming option is not chosen by the patient due to the additional cost, data will not be transmitted immediately, implying additional data delay; otherwise, data can be transmitted immediately, but at higher (i.e. poorer) cost.
- Quality of Evidence: degree of conformance with guidelines, rules of certification/legislation bodies and evidence based medicine (e.g. controlled trials). This is aligned

with [24]. Hence, it indicates how reliable the source of information is. For example, 'poor' Quality of Evidence (QoEvidence) of HR data is defined when an HR sensor does not hold the CE certificate that guarantees output data quality.

Notice that, depending on the context, the impact of the QoD dimensions may vary. For example, if the patient has a fixed cost for roaming data, cost may not have much influence. However, if the patient is abroad, data roaming cost to transmit clinical data to the BE DSS may need to be considered.

In order to compute clinical data quality, using the layering technique [30], we developed a conceptual model which defines the functional and non-functional relation between technological and clinical concepts. The non-functional relation is based on the functional relation and defines relations between QoD of technological variables, such as raw data, and QoD of meaningful clinical variables relevant to the treatment, such as HR. The non-functional relation also includes computational models used to determine the impact of QoS of technological resources on QoD. As described in [15], these computational models consist of transfer functions (f_i), such as mathematical functions or graph-based mapping functions. These transfer functions are used to provide QoD based on QoS and previous QoD: $QoD_i = f_i(QoS_i, QoD_{i-1})$. For example, the input data (D_{i-1}) of a technological resource, such as a HR processor, may be an electrocardiogram (ECG) with Signal to Noise ratio (SNR) equal to 0.7 dB. The SNR is an attribute that characterizes the Accuracy of the ECG. The HR processor manufacturer may have a SNR robustness graph, which shows how robust the HR process is against noise. Based on a graph-based mapping function, we can obtain the Sensitivity (Se) and Specificity (Sp) values of the output HR. From these values and applying a simple mathematical function, the scalar value of accuracy is computed: Accuracy = Se × 0.5 + Sp × 0.5 (e.g. Accuracy = 85 %). Other examples to compute QoD dimensions are shown in [15]. These QoD dimension values need to be translated to a clinically meaningful quality grades by applying the Relative Quality of Data step.

Relative QoD is a relevant concept in QoD that emphasizes the importance of taking into account the consumer's viewpoint to judge QoD based on a "fitness for use" study [26]. In our research, we applied this concept by stratifying the scalar values of the QoD dimensions from the computational models into one of four quality grades: High, Medium, Low, Very Low [15, 30]. These grades are adapted from [24]. This stratification model is based on the medical practitioners' interpretation of the scalar QoD values [15], considering also additional technological information. For example, a scalar value of HR clinical variable with accuracy = 50 % may be due to a noisy ECG signal, where the R peaks of the ECG used to compute HR are not easily identified. Additionally, the medical practitioners considered the context of the application (e.g. outdoors physical exercise treatment) and the user condition (e.g. persistent AF patient) to determine each QoD grade. Hence, with the support of the QoD expert, the medical practitioners determined for each parameter (e.g. HR) the clinically relevant ranges of QoD dimensions (e.g. Accuracy) to be mapped onto each QoD grade in each context. For example, Accuracy = 85 % may correspond to a 'Medium' quality grade in a specific case (Table 1), while in a different context, this value may be mapped to 'Low'. In order to calculate this relative QoD information aligned to the context and user,

Table 1. Stratification model example for HR_{mon} accuracy [15]

Clinical variable HR_{mon}	
Scalar ranges	Grade value
[0 %, 69.9 %]	Very Low
[70 %, 79.9 %]	Low
[80 %, 94.9 %]	Medium
[95 %, 100 %]	High

QoD Broker needs treatment declarative data, which contains the necessary context and user information. This information is provided by the mDSS.

Table 1 illustrates an example of the stratification model for the scalar values of Accuracy QoD dimension of heart rate monitoring (HR_{mon}) clinical variable. In this example, the context is an outdoors physical exercise treatment for permanent Atrial Fibrillation (AF) patients. As shown in Table 1, scalar values ranges, in this case from 0 % to 100 %, are mapped to one of the quality grades as specified by the cardiologist.

QoD Broker acquires treatment declarative data from the mDSS and QoS information and clinical data from sensors and patient GUI to compute QoD information. Additionally, QoD Broker provides technological recommendations to the patient via the GUI (Fig. 2) to improve clinical data quality, so that treatment efficacy and patient safety can be optimized. For example, QoD Broker may ask the patient to re-enter a data value when an error is detected, or it may advise the patient to charge the smartphone battery before physical exercise to pre-empt battery failure during exercise therapy. In this way the QoD Broker not only detects low quality data, but can also, in some cases, pre-empt collection of low quality data and ensure capture of higher quality data. The QoD Broker is implemented in the mobile part of the MobiGuide system to acquire QoS information from the technological resources. The mDSS processes the clinical data and its QoD. This enhances the safety of the mDSS recommendations, since its knowledge is based on the QoD-aware CIG. The QoD information is also stored in the back-end PHR, so that it can be used by the medical practitioner and by the BE-DSS to support a QoD-aware decision making process.

As discussed by Weber et al. [27], a QoD method is needed to design better health information systems. Their study focuses on Data Quality by contract (DQbC), which applies pre-conditions (data input constraints) and post-conditions (assurances of the output data), and compares the data with other data sources to quantify the quality. Our approach focuses on the QoD for an autonomous mobile patient guidance system. However the DQbC design theory and method is applicable in our model once the data is stored in the PHR.

6 The Mobile Decision Support System

The mDSS component is an Android service which communicates with other components by subscribing and publishing to the appropriate channels provided by the BAN service middleware. The mDSS functions as a sort of communication hub within

the mobile device, so a substantial part of the mDSS consists of message handling and passthrough mechanisms, making it a suitable host for the data simulator (see below).

The most interesting part of the mDSS is the projection engine. The projections were developed in several steps. First, semi-formal projections were specified according to the overall BE-DSS CIG specification. These were then developed into fully executable specifications using a projection language that was designed to be simple, yet flexible, powerful and generic. The projection language is based on JavaScript for execution using the Rhino scripting engine, which was chosen for its technical suitability: it runs on Android and enables full processing state save/restore by means of its built-in Continuation mechanism.

High-level functions were developed for enabling concise specification of guidelines. These functions principally operate on the local database, whose entries consist of time stamped events with one or more values and annotations attached to each. The most important functions are the following:

- Annotations. An annotation statement specifies a condition under which an event should be annotated with a particular annotation, or a set of events according to particular conditions within the set.
- Temporal queries. This involves specifying a calculation over time, such as a sum of values or count of events occurring within a specified time window. The projection can be made to wait for (trigger on) a temporal query. To ensure that the system does not re-trigger on the same condition again, the events that led up to the trigger are tagged so that they are no longer considered for the next temporal query. Additionally, a refractory period can be defined that specifies how long the trigger will remain inactive after triggering.
- Calendar queries. In some cases, the system reacts to events in the user's calendar, in particular if risky events like operations are planned in the near future.
- Periodic wait. The system can wait for a particular weekday or time to occur. A start and end date can also be specified.
- Event functions. Events from the database can be queried, manipulated, and stored.
- Message functions. Several functions exist for sending specific types of messages, such as patient notifications, measurement requests, and callbacks.

Apart from the projection engine, the mDSS also contains a summarizer component which summarises the BAN data streams according to the clinicians' requirements, thus mitigating any problem of information overload from the raw data. In the MobiGuide project, it was decided in consultation with the clinicians that summaries are needed for the AF application, specifically for the streaming heart rate and R-R interval data derived from the BioHarness™ sensor. As recommended by the AF guideline [17], patients should wear this sensor regularly during daily living, and whenever an AF symptom is felt, and for each monitoring session, the Summarizer computes the standard deviation of the R-R intervals every minute as well as the average, minimum and maximum heart rate detected during the whole session. Furthermore, the Summarizer receives data concerning episodes of irregular heart rate from the AF detector software running in the BAN and computes from it the total proportion of time in which the heart rate is irregular as well as the average, minimum and maximum heart rate of each irregular heart rate episode.

To enable rigorous testing, we designed a data simulator, which provides a generic means for testing the MobiGuide system against different scenarios and on data spanning potentially long time periods. A simulation scenario is subdivided into multiple steps which can be started from the GUI, allowing the user to interact with the system between steps.

7 Discussion and Conclusions

The components described here have been implemented as part of the MobiGuide system, which is being evaluated against the AF and GDM guidelines. Using a participatory design approach, medical domain experts validated the domain knowledge and system functionality during system design and development. Patient user as well as clinician user participation during the design trajectory was also a priority, with patient focus groups and surveys used to gain feedback from patients on the concepts, the design and the perceived value of the service. Regarding impact of Quality of Data, the medical practitioners understood the potential negative implications of degradations of technological context and determined that the inclusion of data quality awareness has the potential to improve patient safety and treatment effectiveness.

The MobiGuide system components have undergone unit and integration testing as well as a pre-pilot testing phase. The pre-pilot study was performed with volunteers in order to verify that the system functionalities run according to the medical requirements and successful results were obtained. Amongst other things, these tests confirmed the technical feasibility of providing QoD-aware guideline-based decision support to patients via a semi-autonomous system running on their smartphones. Currently, as a further step in the clinical and technical evaluation, the MobiGuide system is being piloted on patients in Spain and Italy. The GDM pilot site is Corporacio Sanitaria Parc Tauli de Sabadell near Barcelona in Spain and the pilot site for AF patients is Fondazione Salvatore Maugeri Clinica del Lavoro e della Riabitazione in Pavia, Italy.

For the University of Twente, the research conducted in MobiGuide together with our partners has extended our research into health BAN applications, amongst others, by incorporating clinical decision support based on clinical guideline knowledge into the BAN application, by distributing decision support functionality between the patient's mobile system and a fixed back-end system via a projection mechanism, and by introducing quality of data awareness to BAN applications.

The research on clinical decision support in the context of evidence-based medicine has produced new modelling approaches to be applied in the analysis of guideline knowledge and generic mechanisms (the projection model and language) for distributing clinical knowledge and decision support functionality. The QoD research demonstrates that the approach applied not only succeeds in detection of data quality problems (thus enabling pre-emption of adverse effects of poor data quality) but also enhancement of clinical data quality through identification and corrective action where certain technological resource problems are identified.

Acknowledgments. The MobiGuide project (http://www.mobiguide-project.eu/) has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 287811.

References

1. Shalom, E., Shahar, Y., Goldstein, A., Ariel, A., Sheinberger, M., Fung, N., Jones, V., Van chooten, B.: Implementation of a distributed guideline-based decision support model in MobiGuide framework. In: Riaño, D., Lenz, R., Miksch, S., Peleg, M., Reichert, M., ten Teije, A. (eds.) KR4HC/ProHealth 2015. LNCS (LNAI), vol. 9485, pp. 111–125. Springer, Heidelberg (2015)
2. Jones, V.M., Bults, R.G.A., Konstantas, D.M., Vierhout, P.A.M.: Healthcare PANs: Personal Area Networks for trauma care and home care. In: Fourth International Symposium on Wireless Personal Multimedia Communications (WPMC) (2001)
3. Jones, V.M., Huis in't Veld, R., Tonis, T., Bults, R.B., van Beijnum, B., Widya, I., Vollenbroek-Hutten, M., Hermens, H.: Biosignal and Context Monitoring: Distributed Multimedia Applications of Body Area Networks in Healthcare. In: Proceedings of the 2008 IEEE 10th International Workshop on Multimedia Signal Processing, (MMSP 2008), pp. 820–825 (2008)
4. Hermens, H., Jones, V.: Extending remote patient monitoring with mobile real time clinical decision support. In: Veltink, P., Eberle, W. (eds.) 4th Annual Symposium of the IEEE-EMBS Benelux. University of Twente, The Netherlands (2009)
5. Jones, V., Hermens, H., et al.: Predicting feedback compliance in a teletreatment application. In: 2010 3rd International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL), pp. 1–5. IEEE (2010)
6. Touati, F., Tabish, R.: U-healthcare system: state-of-the-art review and challenges. J. Med. Syst. **37**, 9949 (2013)
7. Hommersom, A., Lucas, P., Velikova, M., Dal, G., Bastos, J., Rodriguez, J., Germs, M., Schwietert, H.: MoSHCA - my mobile and smart health care assistant. In: 2013 IEEE 15th International Conference on e-Health Networking, Applications Services (Healthcom), pp. 188–192 (2013)
8. Circulation, Journal of the American Heart Association. Guidelines for the Management of Patients With Atrial Fibrillation: A Report of the American College of Cardiology Foundation/American Heart Association Task Force on Practice Guidelines 123, e269–e367 (2011)
9. Rigla, M., Tirado, R., Caixàs, A., Pons, B., Costa, J.: Gestational Diabetes Guideline CSPT. MobiGuide Project Deliverable (FP7-287811) Version 1.0, 12/02/2013
10. Shahar, Y., Miksch, S., Johnson, P.: The Asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines. Artif. Intell. Med. **14**(1), 29–51 (1998)
11. Shahar, Y., Young, O., Shalom, E., Galperin, M., Mayaffit, A., Moskovitch, R., Hessing, A.: A framework for a distributed, hybrid, multiple-ontology clinical-guideline library, and automated guideline-support tools. J. Biomed. Inform. **37**, 325–344 (2004)
12. Shalom, E., Shahar, Y., Taieb-Maimon, M., Bar, G., Yarkoni, A., Young, O., Martins, S.B., Vaszar, L., Goldstein, M.K., Liel, Y., Leibowitz, A., Marom, T., Lunenfeld, E.: A quantitative assessment of a methodology for collaborative specification and evaluation of clinical guidelines. J. Biomed. Inform. **41**, 889–903 (2008)

13. Sacchi, L., Fux, A., Napolitano, C., Panzarasa, S., Peleg, M., Quaglini, S., Shalom, E., Soffer, P., Tormene, P.: Patient-tailored workflow patterns from clinical practice guidelines recommendations. In: MEDINFO 2013, pp. 392–396 (2013)
14. Fung, N.L.S., Widya, I., Broens, T., Larburu, N., Bults, R., Shalom, E., Jones, V., Hermens, H.: Application of a conceptual framework for the modelling and execution of clinical guidelines as networks of concurrent processes. Procedia Comput. Sci. **35**, 671–680 (2014)
15. Larburu, N., Bults, R.G., Widya, I., Hermens, H.J.: Quality of data computational models and telemedicine treatment effects. In: 2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom), pp. 364–369 (2014)
16. Dey, A.: Understanding and using context. Pers. Ubiquit. Comput. **5**(1), 4–7 (2001)
17. Goldstein, A., Quaglini, S., Sacchi, L., Panzarasa, S., Napolitano, C., Larburu, N., Bults, R., Rigla, M., García-Sáez, G.: MobiGuide Deliverable D4.1: CIGs KB (2014)
18. Shalom, E., Shahar, Y., Parmet, Y., Lunenfeld, E.: A multiple-scenario assessment of the effect of a continuous-care, guideline-based decision support system on clinicians compliance to clinical guidelines. Int. J. Med. Inform. **84**(4), 248–262 (2015)
19. Berti, L.: Quality and recommendation of multi-source data for assisting technological intelligence applications. In: Bench-Capon, T.J.M., Soda, G., Tjoa, A.M. (eds.) DEXA 1999. LNCS, vol. 1677, pp. 282–291. Springer, Heidelberg (1999)
20. Berner, E.S.: Clinical decision support systems: state of the art. AHRQ Publication (09-0069), 4–26 (2009)
21. McCormack, J.L., Ash, J.S.: Clinician perspectives on the quality of patient data used for clinical decision support: a qualitative study. In: AMIA Annual Symposium Proceedings, vol. 2012, p. 1302. American Medical Informatics Association
22. Chengalur-Smith, I.N., Ballou, D.P., Pazer, H.L.: The impact of data quality information on decision making: an exploratory analy-sis. IEEE Trans. Knowl. Data Eng. **11**(6), 853–864 (1999)
23. Wand, Y., Wang, R.Y.: Anchoring data quality dimensions in ontological foundations. Commun. ACM **39**(11), 86–95 (1996)
24. Guyatt, G.H., Oxman, A.D., Kunz, R., Falck-Ytter, Y., Vist, G.E., Liberati, A., Schunemann, H.J., GRADE Working Group: Going from evidence to recommendations. BMJ **336**(7652), 1049–1051 (2008)
25. Widya, I., Stap, R.E., Teunissen, L.J., Hopman, B.F.: On the end-user QoS-awareness of a distributed service environment. In: van Sinderen, M., Nieuwenhuis, L.J.M. (eds.) PROMS 2001. LNCS, vol. 2213, pp. 222–237. Springer, Heidelberg (2001)
26. Wang, R.Y., Strong, D.M.: Beyond accuracy: what data quality means to data consumers. J. Manage. Inf. Syst. **12**(4), 5–33 (1996)
27. Weber, J.H., Price, M., Davies, I.: Data quality by contract – towards an architectural view for data quality in helath information systems. In: Prohealth (2015)
28. Ballou, D.P., Pazer, H.L.: Cost/quality tradeoffs for control procedures in information systems. Omega **15**, 509–521 (1987)
29. Widya, I., van Beijnum, B. J., Salden, A.: QoC-based optimization of end-to-end M-Health data delivery services. In: 14th IEEE International Workshop on Quality of Service, IWQoS 2006, pp. 252–260 (2006)
30. Larburu, N., Widya, I., Bults, R.G., Hermens, H.J.: Making medical treatments resilient to technological disruptions in telemedicine systems. In: 2014 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI), pp. 285–288 (2014)

Health Information Systems and Clinical Data

Data Quality by Contract – Towards an Architectural View for Data Quality in Health Information Systems

Jens H. Weber[1,2(✉)], Morgan Price[1,2], and Iryna Davies[2]

[1] Department of Computer Science, University of Victoria, Victoria, BC, Canada
jens@uvic.ca
[2] Department of Family Practice, University of British Columbia, Vancouver, BC, Canada

Abstract. Clinical Information Systems (CIS) have become a pivotal appliance in modern healthcare systems. Their adoption and increasing integration is driven by expectations related to better health outcomes and cost effectiveness. In practice, however, a lack of data quality (DQ) is often referred to as a significant inhibitor, impeding the full realization of these benefits. Although many authors have reported on DQ related problems, attaining and sustaining DQ in CIS has been a multi-faceted and elusive goal. The current literature on DQ in health informatics consists mainly of empirical studies and practitioners' reports. Reports often focus on a particular issue or quality attribute but lack a holistic approach to addressing DQ 'by-design'. This paper seeks to present a general framework for clinical DQ, which blends engineering theories with concepts and methods from health informatics. We define a new architectural viewpoint for designing and reasoning about DQ in CIS. We also introduce the notion of DQ Probes for monitoring and assuring DQ during system operation. Finally, we validate our framework with a real-world application case study.

1 Introduction

It is a well-known fact that the benefits of using any information system are limited by, among other things, the quality of the data maintained within it. Clinical Information Systems (CIS) are no exception, however, the high complexity and diversity of data related to health care processes is known to put the clinical data quality (DQ) problem on a different scale as compared to other business sectors, e.g., banking and finance. The issue of clinical DQ has many facets and, while there is not yet a single commonly accepted standard definition of this concept, there has been growing consensus on a number of aspects that have to be taken into account when discussing DQ in a health informatics context. Weiskopf and Weng have mapped DQ issues reported in the Electronic Health Record (EHR) literature to five quality attributes (*completeness*, *correctness*, *currency*, *plausibility* and *concordance*) [1]. Earlier the IOM defined four quality attributes (*completeness*, *accuracy*, *legibility* and *meaning*) but pointed out that digital record keeping has resolved most legibility related issues traditionally known from paper records [2]. A recently updated briefing by an AHIMA working group

D. Riaño et al. (Eds.): KR4HC/ProHealth 2015, LNAI 9485, pp. 143–157, 2015.
DOI: 10.1007/978-3-319-26585-8_10

defines ten DQ quality attributes, including *accuracy*, *accessibility*, *comprehensiveness*, *consistency*, *currency*, *definition*, *granularity*, *precision*, *relevancy* and *timeliness* [3].

The currently published health informatics literature on these DQ attributes is primarily of empirical nature or focuses on specific practical aspects related to defining, measuring or improving DQ for a particular purpose [4]. Comparably few authors have published on theories and methods for *engineering* CIS for DQ 'by-design', apart from imposing constraints on a CIS database schema (e.g., typing constraints, cardinalities, referential integrity etc.). However, schema constraints address only a limited number of DQ concerns (e.g., concerns related to currency and concordance cannot be addressed) and schemas that are constrained too tightly bear the risk of barring users from recording information – even if it is deemed inconsistent at the time of entry [14, 15]. Theoretical models and principled design-focussed approaches have been promulgated in other domains and the general information system research community, but health informaticians have been reluctant to simply adopt them because their questionable validity for healthcare processes with their unique organizational, behavioural and strategic concerns [4, 5].

We hypothesize that the lack of a fundamental theory and method for engineering CIS for DQ presents an impediment to the design and evolution of better health information system. The purpose of this paper is to define such a theory and method, referred to as *DQ by Contract* (DQbC). DQbC draws on Engineering theory as well as the Health Informatics body of knowledge. We present the general method and provide a concrete example for its application in practice.

The rest of this paper is structured as follows. The next section gives an overview of related work. We define the theory and method for DQbC in Sect. 3, illustrate two application examples in Sect. 4 and report on real-world industrial case study in Sect. 5. We then discuss the current state of DQbC including its limitations (Sect. 6) and close with concluding remarks and pointers to future work (Sect. 7).

2 Related Work

The method used in constructing the theory and method for DQbC is primarily based on a study of the empirical literature on DQ in health informatics and a survey of engineering theories and methods related to information system quality in general and DQ in particular. While there is common agreement that clinical DQ is a multi-dimensional concept, proposed categorizations vary with respect to scope, terminology, and granularity [1–3]. Rather than attempting to further the definition of a standard model of quality dimensions, we set out to develop a theory and method to aid design and evolution of CIS with DQ in focus. We have adopted Weiskopf and Weng's categorization of quality dimensions, since it is supported by a large body of empirical evidence [1]. However, our theory is not limited to this particular categorization but equally applies to other conceptualizations of clinical DQ.

The DQbC method is based on the engineering paradigm of *functional decomposition*, in which complex system behaviour is broken down into more elementary discrete functions [6]. We suggest that a functional perspective for designing for and reasoning about DQ properties of CIS is more appropriate than an alternative

decomposition paradigm, e.g., data-oriented or object-oriented decomposition, because DQ concerns must be modelled *in context* of the particular use cases (functions) that consume or provide the data. The idea of treating data as a functional *product* is not new but is foundational to general information quality methods, e.g., IP-MAP [18]. What is new in our current paper is the notion of data contracts, which has been inspired by the software engineering method of *Design by Contract* (DbC) [7]. The DbC approach was developed in the domain of component-based software engineering as a way to formalize and monitor the expectations and guarantees of the interfaces among software components. In this approach, software interfaces are equipped with '*contracts*', consisting of pre-conditions (expectations on how the component should be invoked to function correctly) and post-conditions (assurances on what a component can be expected to perform *if* invoked correctly).

The classical DbC paradigm has only limited applicability to the DQ problem in CIS. We therefore extended the classical DbC concept to enable quality assertions over DQ concerns related to the canonical types of health information sources in health information system architectures, including EMRs, EHRs (shared health records), other types of CIS, and public health research databases [8]. We introduced and defined the concept of *DQ Probes* as a way to make DQ related assertions in pre- and postconditions associated with CIS data processing functions. Our theory and method has been applied to several CIS data projects, including a project on building a data sharing and analytics network for primary care clinics [9].

3 Data Quality by Contract

3.1 A Functional Theory for Reasoning About Clinical DQ

Quality is generally defined as 'fitness for *use*' or 'fitness for *purpose*' [4]. It makes no sense to discuss the quality of a given unit of data without framing the context of its intended use case. A broad spectrum of different types of fitness criteria may be applicable (see previous section). These criteria are commonly referred to as *quality attributes* and include absolute criteria (e.g., the existence of a particular datum) as well as relative degrees of fitness (e.g., the precision of a datum). The degree to which a datum meets a certain quality attribute may be associated with that datum in form of *meta-data* (e.g., data on the currency or provenance of the actual clinical data), or it may be inferred based on a process that uses any combination of other contextual data sources, e.g., population health statistics may be used to evaluate the plausibility of clinical patient data recorded in an EMR.

From a conceptual point of view, any *use case* of data can be modelled as a *function* that maps a certain domain of input data (including meta-data) to a particular range of output data (including meta-data on the output). Note that this theoretical model by no means limits the use of data to automated (machine-based) processes but also applies to the use of data by human actors or a combination of human and machine-based processing. We also note that a conceptual function (even an automated one) may not necessarily be implemented within a single software component (i.e., the implementation of a conceptual function may cross-cut system components).

Clearly, the quality of the input data (as described in the input meta-data) impacts the reliability of the function's output (as described in its output meta-data). Moreover, functions may have varying degrees of *inherent reliability*, which also impacts the quality of its result, e.g., the process of a radiologist reading a medical image may have a specific error rate, or an automated algorithm for screening images for abnormalities may have certain accuracy.

Data use cases (functions) have *constraints* on the quality of the data required on their input (preconditions). In cases where the input data meets those pre-conditions, such functions may *assure* certain quality attributes about their output data (post-conditions) (cf. Fig. 1).

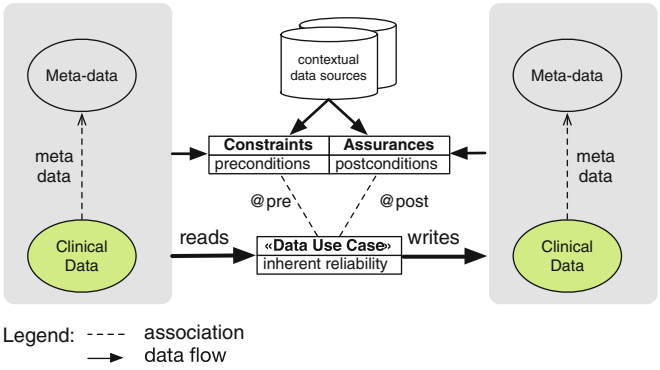


Fig. 1. Functional model for reasoning about DQ

Constraints and assurances of data use cases are rarely modelled explicitly in current CIS design practices. In this paper, we suggest that they should, since doing so allows CIS engineers to reason about DQ problems during design and implementation and pinpoint where they arise: A DQ problem arises in any data use case that does not meet its post-conditions while all of its pre-conditions have been met.

3.2 Reasoning About DQ

The functional theory described above is a sufficient abstraction to model all data processing activities in health information systems in a way conducible to reasoning about DQ related concerns (and designing systems for DQ). In order to see this we would like to make two clarifying remarks: Firstly, we point out that the notion of *data* in the above model is not limited to electronic data, but also encompasses data communicated or stored in other forms, e.g., data communicated orally in a patient interview. The second remark pertains to the modelling of the 'emergence' of data from health care related acts that observe phenomena in the physical world, (e.g., clinical observations, lab tests, etc.) In our theory, these are modelled as functions that generate data (without necessarily consuming input data in the process). In practice, however, virtually all such '*observational'* functions have some input data, even if it is limited to

the identity of the patient they are applied to. (Indeed patient-misidentification is one of the most prevalent apparent root causes for DQ-related safety accidents involving clinical information systems [10].)

The above theory can be used as an analytical tool to detect and reason about DQ issues in CIS. In order to do so, all data use case functions of interest are defined along with their constraints (pre-conditions) and assurances (post-conditions). DQ concerns can then be analyzed by composing *DQ flow views* that focus on producers and consumers of clinical data of interest and align their respective assurances and constraints. Figure 2 shows a simple example involving a use case (automated) that generates chronic disease management care reminders for patients with Diabetes. The use case requires the active problem of Diabetes to be encoded (in SNOMED CT - SCT) in the patient's list of active problems. Moreover, it requires A1C lab test results to be encoded (in LOINC) and to include the most recent result on the patient. Our example flow model shows that the use cases that *generate* the required data do not assure all the required quality attributes of the "Care Reminders" use case. The "Record diagnosis" use case generates data in coded and/or unstructured form. The "Import lab" use case assures LOINC but provides no assurance on the currency of the data. (For simplicity we omit any preconditions of the data producing use cases and the assurances of the data-consuming use case in Fig. 2).

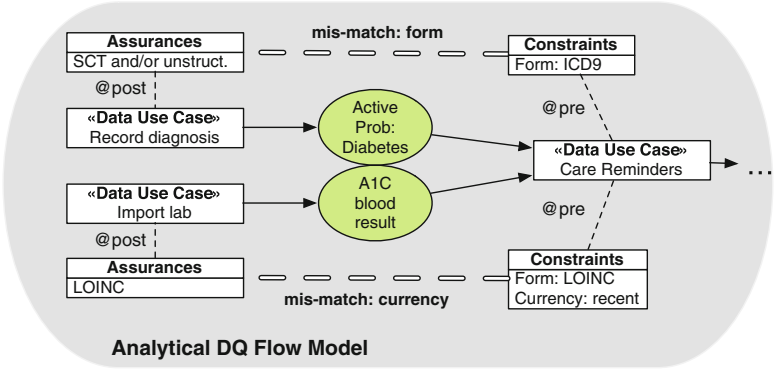


Fig. 2. Example for an Analytical DQ Flow Model

3.3 DQ Flow Modelling During System Design

We suggest that DQ Flow modelling (as shown above) is an excellent analytical tool for *designing* CIS for DQ concerns. While this claim still needs to be validated empirically, the proposed method is similar to the *Design-by-Contract* (DbC) approach, which has been used effectively in software engineering best practices today [7]. Indeed, the set of constraints and assurances associated with a use case function can be considered the function's "contract", which promises certain quality attributes about the function's output data, given that its input data meets certain pre-conditions.

DQ Flow Models provide a novel DQ-focussed architectural viewpoint for system analysis and design [6] and is in line with the documentation philosophy described in the ISO standard 42010:2011, which prescribes documenting architectural view points according to stakeholder quality concerns.

In classical software engineering, DbC is not only an architectural design method but also an approach to *implementing* software components, so that the correctness of their composition can be *monitored* [7]. In this approach, pre-conditions of computational functions are implemented as *assertions* that are checked when that function is called and its post-conditions (assurances) can be implemented as assertions that are validated after the function has executed.

Unfortunately, the classical DbC model known from software engineering has only limited applicability to monitoring DQ properties in CIS. It has been intended to monitor the interaction of software components and is not by itself suitable for monitoring data (flow) quality in (clinical) information systems, which involve automated as well as manual functions. Apart from the fact that human processing functions are considered "inside the box" in CIS, another key difference is that DQ concerns are broader in scope than the traditional concerns of program correctness. Still, an extension of the DbC paradigm is possible – and we will suggest one below.

3.4 DQ Monitoring with DQ Probes

The classical notion of DbC contract monitoring can be extended, to fit the CIS problem domain. We do so by introducing the concept of *data quality probes*. In classical DbC, a function invocation $(o_1,\ldots,o_n) := fun(i_1,\ldots,i_m)$ can define pre-conditions that make logical assertions about the state of its input parameters $@pre(i_1,\ldots,i_m)$ and post-conditions that make assertions about the state of its input and output parameters $@post\ (o_1,\ldots,o_n,\ i_1,\ldots,i_m)$. We extend (and generalize) this mechanism by introducing a set of special side-effect-free predicate functions, referred to as *DQ Probes*. The general signature of a DQ Probe for a given use case function takes the form Q :D x … → BOOL, where D is the set of input/output data of that function and the ellipses represents a placeholder for additional arguments that may be used depending on the particular type of DQ probe. We define five types of DQ probes, which are used to address five conceptual classes of DQ concerns (Type 0–4). In this extended DbC paradigm for clinical DQ concerns, pre-conditions and post-conditions are defined as logical conjunctives over applications of DQ probes, i.e., $@pre(Q_1,\ldots,\ Q_y)$ and $@post(Q_1,\ldots,\ Q_z)$, respectively.

Intrinsic DQ Concerns (Type 0). The most basic DQ concerns directly target the state of data items, i.e., concerns that can be detected solely by evaluating the state of the data itself. We refer to this class of DQ concerns as *intrinsic* (*Type 0*). Intrinsic DQ concerns address a specific class of *correctness* quality attributes (cf. [1]), namely those related to ***conformance***. An example for a Type 0 DQ concern is the use of a particular format or code in a data field, e.g., a properly formatted date field *DD:MM:YYYY*, the use of a code from a certain reference set of terms, or even the simple existence of a given data item.

DQ probes for monitoring intrinsic concerns take the form Q :D → BOOL, i.e., they focus solely on evaluating the conformance of a set of given data items.

Intrinsic-Meta DQ Concerns (Type 1). Another type of concerns pertain to *meta-data* associated with the actual data parameters of a given function, evaluating the ***provenance*** of that data, e.g., information about *where* the data originated, the procedure of *how* the data was obtained, the *currency* of the data, etc. We refer to this class of DQ concerns as *intrinsic-meta* (Type 1). DQ probes for monitoring Type 1 concerns take the form Q :DxM → BOOL, where M denotes the set of quality meta-data associated with the items in D (Fig. 3).

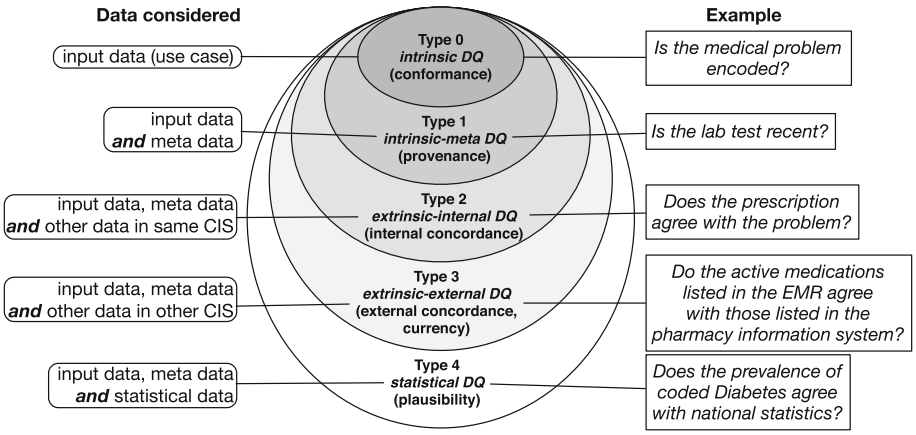


Fig. 3. Classification of DQ Probes

Extrinsic-Internal DQ Concerns (Type 2). The types of DQ probes discussed so far are still limited with respect to the type of DQ validations they can perform. For example, aspects of ***concordance*** of data items with respect to *other* data in the CIS cannot be validated (cf. [1]). Conceptually, we differentiate concordance aspects into two classes of concerns. The first class concerns concordance with other data *internal* to the same CIS, i.e., other data that is under the control of the same clinic or organization. Examples for such concerns are the concordance of a particular data (e.g., insulin medication in the patient's medication list) with other data in the CIS (e.g., the presence of Diabetes in the same patient's list of active medical problems). We refer to these DQ concerns as *extrinsic-internal* (or Type 2) concerns. DQ probes for monitoring Type 2 concerns take the form Q :D x M x B → BOOL, where B denotes the entire database associated with the particular CIS.

Extrinsic-External DQ Concerns (Type 3). A second class of concordance-related DQ concerns focus on validating consistency of data with respect to data that resides *external* to the CIS, i.e., data that is not under control of the clinic or organization. This may include other CIS (at other clinics) that interoperate with the current CIS as well as other information systems used in the integrated healthcare system, e.g., patient registries,

laboratory information systems, pharmaceutical systems, personally-controlled health records, etc. In order to validate these kinds of DQ concerns, we need DQ probes to be able to access data in these external systems or allow for comparison checks in some manner (e.g. when data is transferred through messaging). We refer to these DQ concerns as *extrinsic-external* (Type 3) concerns. Type 3 DQ probes take the form Q :D x M x I → BOOL, where I denotes a set of interoperable (external) databases in the health information infrastructure. Note that Type 3 DQ probes are not limited to validating ***(external) concordance*** but are also required to validate ***relative currency***, e.g., to validate that a particular lab test is indeed the most recent one performed for a patient.

Statistical DQ Concerns (Type 4). Finally, statistical DQ concerns address the ***plausibility*** dimension in Weiskopf and Weng's categorization of quality attributes [1]. Their validation requires a fifth and final class of DQ probes that check a use case function's data against statistical data distributions in population health information systems in order to validate plausibility (or meaningfulness) of those data. For example, a Type 4 DQ probe may evaluate the prevalence of recorded Diabetes cases in a CIS against population health statistics. DQ probes for monitoring Type 4 concerns take the form Q :D x M x S → BOOL, where S denotes a statistical database on population health data.

4 Deploying DQ Probes in Practice

DQ probes can be deployed in practice when designing or improving data use case functions. We discuss two examples in this section. The first example illustrates the use of DQ probes in pre-conditions (constraints) of a data use case function, while the second one illustrates the application of DQ probes for validating post-conditions (assurances) of a data use case.

4.1 DQ Probes for Constraining Use Case Input (Pre-Condition)

The first case involves the use of CIS data to answer a clinical research question on the rate of polypharmacy in the elderly as documented in primary care clinics' EMRs. Figure 4 shows an example data use case called "*Polypharmacy Study*", which deploys five DQ probes, one of each type. For this simple example, we define polypharmacy as situations where a patient is currently taking five or more medications.

The pre-conditions of the polypharmacy data use case include DQ probes of all five types. The first DQ probe (Type 0) requires that a sufficient number of medication data entries are coded with the appropriate coding system (DIN). The second DQ probe (Type 1) considers meta-data associated with patients and considers only *active* patients (i.e., archival records of former patients are excluded). The third DQ probe (Type 2) constrains participating CIS to those ones that usually encode prescriptions with DIN codes, e.g., with less then 5 % of un-conformant medications. The fourth DQ probe considers EMRs from other GP clinics in the research network to ensure that individual patients are accounted for only once. (The number of patients listed as 'active' in multiple practices must be lower than 5 %.) Finally, the fifth DQ probe validates the

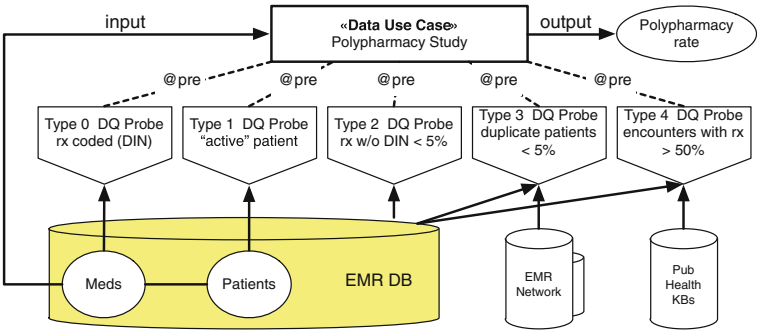


Fig. 4. DQ Probes in pre-conditions of Data Use Cases

plausibility of the overall prescription encoding practices in each clinic with public health knowledge bases, which indicate that, on average, approx. 50 % of encounters result in at least one medication prescribed [11].

4.2 DQ Probes for Assuring Use Case Output (Post-Condition)

Our second example discusses the use of DQ probes to assure quality attributes related to the *output* of data use cases. To some extent, data validation functions associated with CIS uses cases is nothing new. Many CIS input forms have data validation functions that are checked upon data entry. However, our theoretical framework provides a fresh categorization of such functions, which spans all aspects of DQ concerns. Most DQ validation functions implemented in current CIS are at the level of Type 0 (intrinsic) DQ probes, e.g., checking for presence of a data element, conformant formatting or coding of the output of data use cases. Figure 5 shows an example for the data use case of electronic prescribing.

Type 1 DQ concerns are less frequently validated in CIS. Figure 5 shows an example that ensures that all recorded medications have been properly signed and ordered (assuming an electronic prescription order system). Type 2 DQ probes ensure concordance of the output data with the rest of the CIS database and, in the context of prescribing, are often part of decision support system, e.g., checking prescriptions against adverse drug interactions, medication interactions, diagnoses, current lab test results, etc.

Type 3 DQ probes require the existence of an integrated health information exchange infrastructure and validate system-wide concordance and currency, e.g., the consistency of a prescription order with other recent data (lab test, prescription, observation) of a given patient. Finally, Type 4 DQ probes validate output data against public health best practice evidence. These functions are often part of electronic guideline and decision support system functions. An example would be the validation of the appropriateness of a particular prescription for a certain demographics [12].

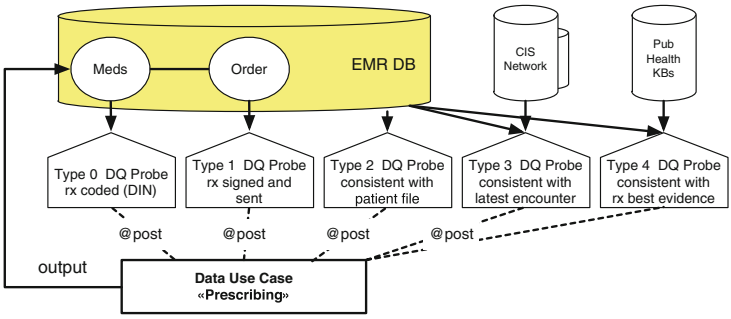


Fig. 5. DQ Probes in post-conditions of Data Use Cases

5 Application Case Study: Physicians Data Collaborative of BC

We have applied and validated the DQbC framework and method presented in this paper in the context of an industrial scale case study: the Data and Sharing and Analytics (DSAS) software of the Physicians Data Collaborative of British Columbia (PDC) [9]. DSAS is an emerging networked software service that connects to a growing number of primary care EMRs in the Canadian province of British Columbia. DSAS allows primary caregivers to share aggregate clinical data for the purpose of quality improvement and research. The DSAS software has adopted Mitre's hQuery architecture [17] that implements a distributed "hub and spoke" query model where EMR data is queried in place and each clinic's results are forwarded and combined at a centralized query hub (cf. Fig. 6).

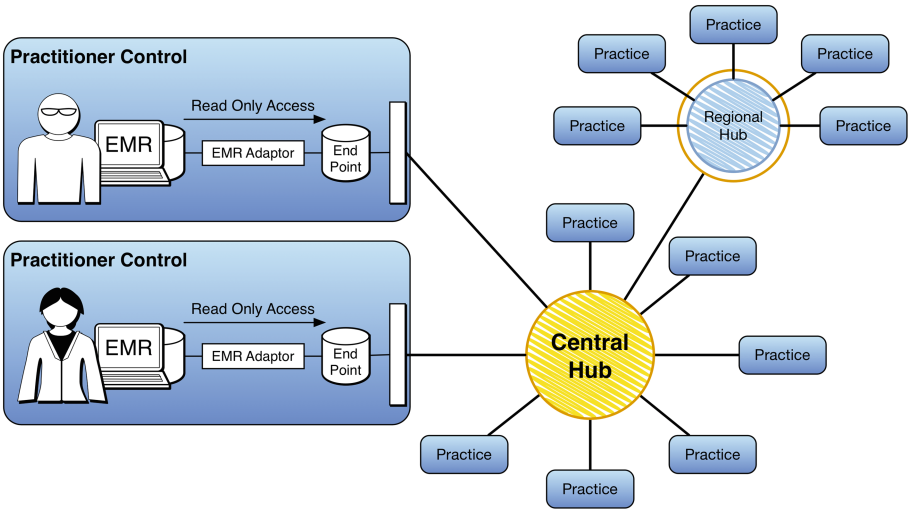


Fig. 6. Hub and Spoke Architecture of the PDC's DSAS Software

The DSAS model is generic and supports many different data use cases. Our lab (funded in part by the PDC) has been developing the software for five data use cases to date, involving a large set of data quality probes. These use cases focus on sharing and analyzing data with respect to medication management, patient-physician attachment, end of life care, and general population health indicators. In the remainder of this section, we will be focussing on the first of these uses cases, which analyzes clinics with respect to their medication management practices, using the STOPP criteria, a complex set of guidelines for screening older persons for inappropriate prescriptions [16]. At present time, this use case involves 32 prescribers.

The current implementation of the STOPP data analysis use case includes 44 of the STOPP indicators, which are computed based on data in the connected EMRs, primarily the patient medication lists, active problem lists, and patient demographic data. The DQ in the connected EMRs varies greatly. Some clinics make full use of their EMR encoding and charting capabilities, while others still rely on natural language notes and largely use the EMR as an "electronic paper record". The STOPP use case function computes indicators for *all* connected clinics, but only indicators computed on input data with sufficient DQ are relied upon for data aggregation and analysis. These indicators are referred to as *sentinel indicators*. We are applying our DQbC method to formalize and implement the above notion of "sufficient DQ".

Figure 7 shows the design of the corresponding data use case *STOPP clinic query*. The use case requires the problem list, medication list and demographics data as input (from each EMR in the network) and computes a set of results for the STOPP indicators as output. The use case has a contract with a precondition that relies on eight DQ probes and a post-condition that assures that the queried clinic is considered a sentinel clinic if the pre-conditions are met (represented by meta-data on the computed STOPP indicators). The three intrinsic (type 0) DQ probes ensure that medications and problems are coded and demographic data is well-formed (e.g., the date of birth). A type 1 DQ probe requires that the considered patient records are flagged as *active*. Since *active* status is the default in EMRs and patient inactivation requires manual intervention (which may not happen consistently), another DQ Probe (type 2) ensures concordance of the *active* flag with the data in the appointment scheduling list. (Patient must have presented in recent two-year period.)

A second type-2 DQ probe ensures concordance between medications (rx) and problem list (upper part of Fig. 7). The implementation of this probe tests for common patterns, such as the existence of insulin in medications and the existence of Diabetes in problems, etc.

The type-3 DQ probe ensures concordance of the medication (rx) list with an external provincial pharmacy information system (Pharma IS). That probe has been designed conceptually but it has not been implemented yet.

Finally, the type 4 probe ensures the statistical plausibility of the data represented in the queried EMR with data available from population health statistics. This probe is implemented as a set of queries that compare prevalence of listed problems, medications, demographics with population health statistics.

Data use cases (such as the STOPP clinic query) as well as DQ probes are implemented as queries on the EMR. More precisely, the queries are run on a virtual medical record database, which has been extracted from the actual EMR in order to

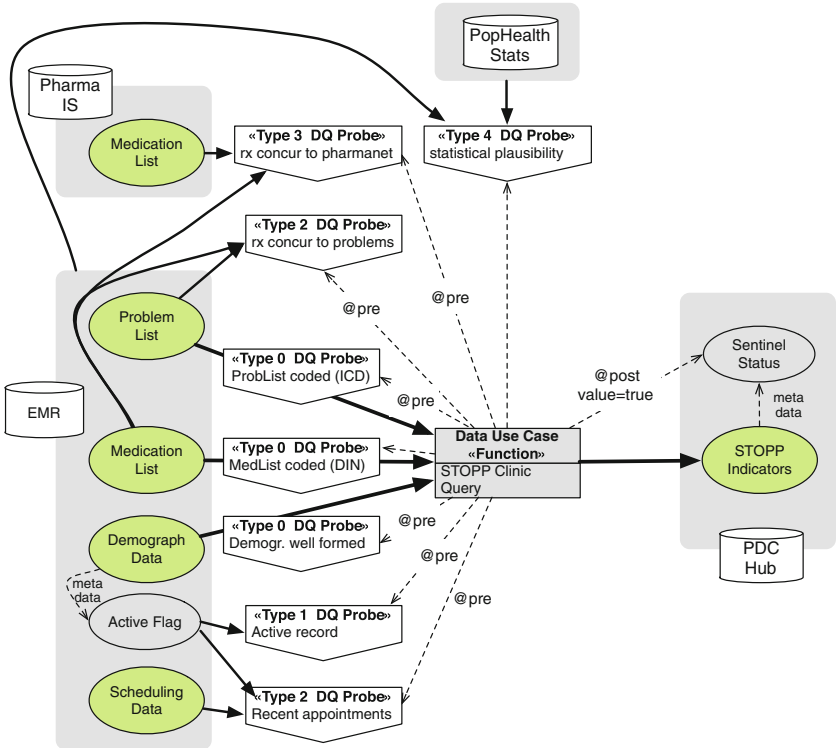


Fig. 7. Data use case: STOPP Clinic EndPoint Query

resolve heterogeneity. However, the details of this middleware are out of scope of this paper and have been reported elsewhere [9]. One important difference between the DQ probe queries and the use case queries is that the DQ probes are required to produce a Boolean value, while the use case queries produce arbitrary (but specified) output data. Internally, though, the DQ queries implemented in our system produce a quantitative measure (e.g., the percentage of coded prescriptions, the percentage of 'active' patients with a recent appointment, the percentage of prescriptions concordant with listed medication problems, etc.). That quantitative measure is then compared to a defined threshold in order to compute the Boolean result that determines whether or not the required condition is considered satisfied. The quantitative measure can be - and in the case of our application – *is* reported to the end user to provide feedback on missing DQ and inform DQ improvement efforts.

6 Discussion

Clinical DQ is a complex and multi-dimensional concept that impacts the realization of benefits of clinical information systems. Different categorizations of clinical DQ dimensions have been proposed in the health informatics literature. While we mainly

orientate ourselves towards Weiskopf and Weng's categorization (rooted in empirical evidence) [1], our DQ design theory and method is applicable to other quality models. We primarily focus on *how* to design (and monitor) systems for quality, rather than on *what* DQ attributes should be considered. The taxonomy of DQ concerns introduced in this paper is driven by architectural system considerations (i.e., which data sources need to be validated) rather than on questions about the purpose of this validation (i.e., the actual quality attribute to be validated). Consequently, our method should be generalizable. Our theory and method covers all of Weiskopf and Weng's quality dimensions as well as AHIMA's quality categories [3] and the IMO DQ model [2], except for legibility, which has little meaning in digitized information management.

Our contribution is different from other DQ-oriented models in health informatics in the sense that it seeks to provide a method and guidance to design CIS systems for data quality. In this sense, it blends architecture engineering methodology with health informatics. Nevertheless, our method is also different from traditional engineering approaches, as it does not solely focus on the technical solution domain. Rather, the concept of a data use case in this paper is not limited to software but may indeed be a human process or a blend between human interactions and machine-based actions.

While this paper is primarily written as a conceptual contribution to the health informatics and engineering methodology, we have begun to validate our method in several real-world industrial projects. The main concepts presented in this theoretical work have emerged from our experiences with CIS implementation projects, including the notion of different types of DQ probes, which we have been implementing in practice for several years. As such, the presented categorization of DQ probes can be seen as a catalogue of DQ-related *design patterns*, referring to proven solutions to recurring problems in a particular domain [6]. We suggest that the presented design method and conceptualization will aid CIS designers with modelling, evolving and improving CIS for DQ concerns. One issue of practical importance that we have not addressed in this paper is how to choose the thresholds for the defined (Boolean) DQ probes. We found that the selection of these thresholds is very much dependent on the particular application (data use case function), which makes it difficult to define a general guideline.

7 Conclusion

Data quality (DQ) in health information systems is a product of processes in the technical domain as well as processes in the organizational ('people') domain. Current approaches to attacking DQ related problems in healthcare primarily focus on the organizational domain, e.g., by increasing user training and putting in place organizational procedures for validating and cleaning data. From a technical perspective, DQ is mainly treated as an emergent phenomenon of a system, which (to a certain degree) can be measured and assessed with computational means. However, treating DQ as an afterthought in CIS design and implementation misses important opportunities for identifying and addressing DQ-related issues 'at the root', during the design, implementation and evolution of CIS. New DQ-focussed design and implementation methods are needed to meet this opportunity. Our research attempts to respond to this need.

This paper presents a theory and foundational method to be used in CIS design with a view to DQ properties. Our current and future work is on further empirical validation of this method and on constructing tool support for facilitating its efficient use in practice. Our current experimentation revolves around the OSCAR open source EMR system [13] and the PDC data sharing and analytics network [9].

References

1. Weiskopf, N.G., Weng, C.: Methods and dimensions of electronic health record data quality assessment: enabling reuse for clinical research. J. Am. Med. Inform. Assoc. **20**(1), 144–151 (2013)
2. Dick, R.S., Steen, E.B., Detmer, D.E., et al.: The Computer-Based Patient Record: An Essential Technology for Health Care. National Academies Press, Washington, DC (1997)
3. Bronnert, J., Clark, J.S., Cassidy, B.S., Fenton, S., Hyde, L., Kallem, C., Watzlaf, V.: Data quality management model (updated). J. AHIMA **83**(7), 62–67 (2012)
4. Mettler, T., Rohner, P., Baacke, L.: Improving data quality of health information systems: a holistic design-oriented approach. In: Proceedings of European Conference on Information Systems, Galway, Ireland (2008)
5. Weber-Jahnke, J.H., Price, M., Williams, J.: Software engineering in health care: Is it really different? and how to gain impact. In: 2013 5th International Workshop on Software Engineering in Health Care (SEHC), pp. 1–4 (2013)
6. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice. Addison-Wesley Professional, Boston (2003)
7. Meyer, B.: Design by contract and the component revolution. In: International Conference on Technology of Object-Oriented Languages, pp. 515–515 (2000)
8. Giokas, D.: Canada health infoway - towards a national interoperable electronic health record (EHR) solution. In: Studies in Health Technology and Information, pp. 108–140 (2005)
9. Price, M., Weber, J., McCallum, G.: SCOOP – the social collaboratory for outcome oriented primary care. In: Proceedings of IEEE International Conference on Healthcare Informatics (ICHI), p. 6. IEEE Computer Society, Verona (2014)
10. Weber-Jahnke, J.H.: A preliminary study of apparent causes and outcomes of reported failures with patient management software. In: Proceedings of 3rd International Workshop on Software Engineering in Health Care (SEHC), Co-Located with ICSE. ACM Press (2011)
11. Ferrell, C.W., Aspy, C.B., Mold, J.W.: Management of prescription refills in primary care: an Oklahoma physicians resource/research network (OKPRN) study. J. Am. Board Fam. Med. **19**(1), 31–38 (2006)
12. Clyne, B., Bradley, M.C., Hughes, C.M., Clear, D., McDonnell, R., Williams, D., Fahey, T., Smith, S.M.: Addressing potentially inappropriate prescribing in older patients: development and pilot study of an intervention in primary care (the OPTI-SCRIPT study). BMC Health Serv. Res. **13**(1), 1–12 (2013)
13. Ruttan, J.: OSCAR. In: The Architecture of Open Source Applications, vol. II: Structure, Scale and a Few More Fearless Hacks (2012)
14. Weber, J.H., Cleve, A., Meurice, L., Ruiz, F.J.B.: Managing technical debt in database schemas of critical software. In: Sixth International Workshop on Managing Technical Debt (MTD), pp. 43–46. IEEE (2014)

15. Balzer, R.: Tolerating inconsistency [software development]. In: Proceedings of the 13th International Conference on Software Engineering, pp. 158–165. IEEE (1991)
16. Gallagher, P., O'Mahony, D.: STOPP (Screening Tool of Older Persons' potentially inappropriate Prescriptions): application to acutely ill elderly patients and comparison with Beers' criteria. Age Ageing **37**(6), 673–679 (2008)
17. Diamond, C.C., Shirky, C., Mostashari, F.: Collecting and sharing data for population health: a new paradigm. In: Health Aff. (Millwood), March 2009
18. Shankaranarayanan, G., Wang, R.Y., Ziad, M: IP-MAP: representing the manufacture of an information product. In: Proceedings of Information Quality, pp. 1–16 (2000)

Author Index