Towards Generic MDE Support for Extracting Purpose-Specific Healthcare Models from Annotated, Unstructured Texts

Pieter Van Gorp[1], Irene Vanderfeesten[1], Willem Dalinghaus[1], Josh Mengerink[1], Bram van der Sanden[1], and Pieter Kubben[2]

[1] School of Industrial Engineering, Eindhoven University of Technology
{p.m.e.v.gorp,i.t.p.vanderfeesten}@tue.nl
[2] Department of Neurosurgery, Maastricht University Medical Center
pieter@kubben.nl

Abstract. Once healthcare-specific models have been captured formally (i.e., in a metamodel-based language), the application of model transformation, analysis and code generation techniques is rather straightforward. Unfortunately, in many healthcare settings valuable domain knowledge is hidden in unstructured text (e.g., in a research paper or a national report on clinical guidelines). This motivates the need for tools to annotate such texts with metadata. Such tools can be prototyped easily for one type of healthcare artifacts (e.g., for clinical guidelines or care pathways) and one purpose (e.g., for workflow management or decision support) but it is a research challenge to build a robust and generic (i.e., metamodel-independent) tool for this important type of model extraction support. This paper desribes our ongoing work to building such a tool on top of a state-of-the-art MDE platform.

1 Introduction

Governments, insurance organizations, hospital boards, physicians and patient organizations support the relevance of rigorous engineering methods for the development and certification of information systems in healthcare. Model-Driven Engineering (MDE) techniques are particularly strong at separating medical and organizational concepts from system implementation details. This is important since information system architectures vary significantly within and between care institutions while at the conceptual level patients cross the institutional and system boundaries. As in most other engineering disciplines, models in MDE are simplified representations that enable one to reason more easily about complex issues. The distinguishing factor is that MDE techniques can combine multiple modeling languages (and formalisms).

MDE leverages explicit modeling language definitions (called metamodels) and model transformation definitions to break down complex modeling problems in more manageable subproblems. We have recently demonstrated that MDE technology is particularly mature in support for generating powerful model

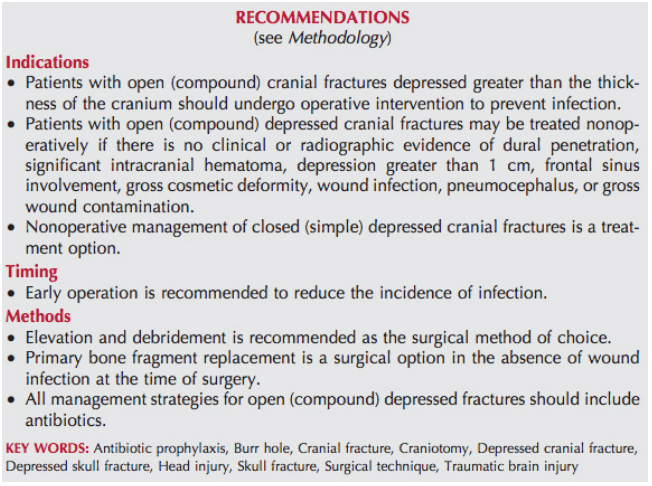
J. Weber and I. Perseil (Eds.): FHIES 2012, LNCS 7789, pp. 213–221, 2013.

editors from annotated metamodels [1]. This short paper focuses on providing novel support for extracting models from unstructured texts.

The remainder of this paper is structured as follows: Section 2 presents a typical example of a clinical decision support (CDS) system. Section 3 presents our solution in two steps (first specific to the CDS example and then more generally). Section 4 describes related and future work and Section 5 concludes.

2 Example Clinical Decision Support System

Fig. 1 shows a clinical guideline (CG) supported by the Congress of Neurological Surgeons and the American Association of Neurological Surgeons. The selected guideline is that for surgical management of depressed cranial fractures.



RECOMMENDATIONS
(see *Methodology*)

Indications

- Patients with open (compound) cranial fractures depressed greater than the thickness of the cranium should undergo operative intervention to prevent infection.
- Patients with open (compound) depressed cranial fractures may be treated nonoperatively if there is no clinical or radiographic evidence of dural penetration, significant intracranial hematoma, depression greater than 1 cm, frontal sinus involvement, gross cosmetic deformity, wound infection, pneumocephalus, or gross wound contamination.
- Nonoperative management of closed (simple) depressed cranial fractures is a treatment option.

Timing

- Early operation is recommended to reduce the incidence of infection.

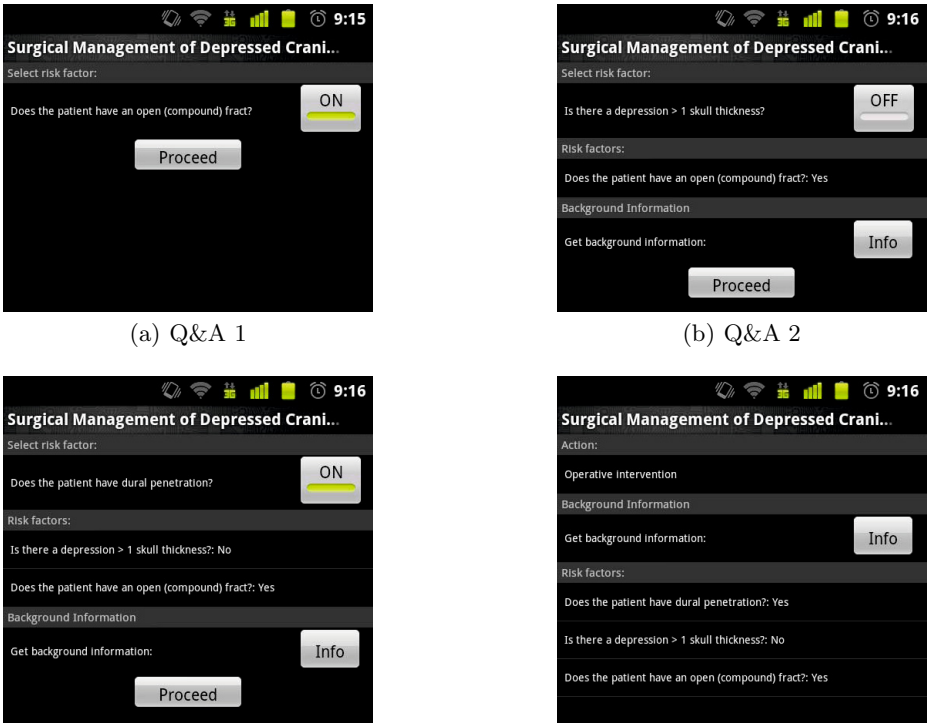
Methods

- Elevation and debridement is recommended as the surgical method of choice.
- Primary bone fragment replacement is a surgical option in the absence of wound infection at the time of surgery.
- All management strategies for open (compound) depressed fractures should include antibiotics.

KEY WORDS: Antibiotic prophylaxis, Burr hole, Cranial fracture, Craniotomy, Depressed cranial fracture, Depressed skull fracture, Head injury, Skull fracture, Surgical technique, Traumatic brain injury

Fig. 1. Example of a Clinical Guideline recommendation (summary based on [2])

As a concrete example of a CDS system derived from such a guideline, consider Fig. 2. That figure shows screenshots of an app that enables specialists to (1) lookup a guideline from a catalogue and (2) retrieve recommendations by entering patient-specific information. Fig. 2(a) shows the app after selecting our example guideline. In our example scenario, the user answers respectively "yes", "no", "yes" (cfr., the "on", "off", "on" buttons) on the series of questions shown on Fig. 2(a) to 2(c). According to the guideline, this leads to the suggestion that there is evidence in favor of performing an early operation (Fig. 2(d)). The underlying decision algorithm is not directly visible in the guideline text. Some medical papers do provide flowcharts to make the proposed decision making process more explicit. However, specialists in a concrete care facility typically still have to adapt such flowcharts to their specific situation. This paper starts from our collaboration with a Dutch academic hospital.



(a) Q&A 1 (b) Q&A 2

(c) Q&A 3 (d) Suggestion

Fig. 2. Example execution of the generated app running on an Android smartphone

In this hospital, one of the neurosurgeons maintains a set of flowcharts to formalize a set of specialized guidelines. The neurosurgeon also programs CDS support in apps such as the one from Fig. 2. These apps are quite popular in both the Android and iOS app stores [3]. Remarkably, the flowcharts are just informal documentation for these apps. We observed that by using MDE techniques, the apps could be generated automatically. That could reduce the development effort and the risk for inconsistencies. It would require however the use of a flowchart editor with a custom metamodel (e.g., with support for modeling CDS questions and links back to medical evidence).

This paper focuses on a key limitation of MDE and our suggested way to overcome it. Our extended experience report also clarifies to Health IT practitioners what existing MDE tools can offer them in the first place [1].

3 Deriving Models from Annotated, Unstructured Text

Fig. 3 sketches our proposed tool-chain from an end-user perspective: first, medical specialists annotate scientific CGs. This can happen in the context of their

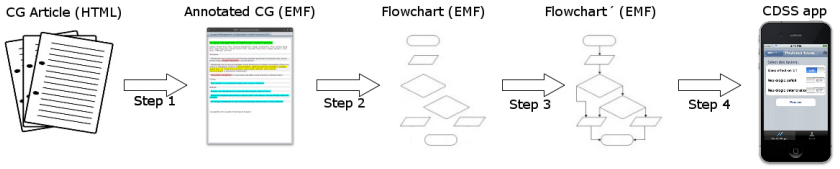


Fig. 3. Model-driven, evidence-based, development of CDS apps

personal continuous learning process or in the context of regularly planned literature review cycles within a hospital. In this step, annotations should be stored in a computer-interpretable form. Second, the guideline annotations are transformed automatically into a flowchart skeleton model. Third, the flowchart is manually refined. Finally, the flowchart is transformed automatically into a CDS app. Configuration files for a more heavyweight CDS system could be generated too but this is not implemented at the time of writing. In the following, we first demonstrate a metamodel-specific tool-chain that we have used to better understand the above workflow in the context of CG-based CDS. Then, we describe our ongoing efforts to derive similar tool-chains more efficiently in the large.

3.1 Ad-Hoc Support: Extracting Flow-Charts from CGs

Fig. 5 shows the text from Fig. 1 within the annotation tool. The title annotation is shown in green. The parts of the text that are considered observations have been annotated in yellow, the actions/treatments are shown in red, and the explanatory elements are shown in blue. The bottom left of Fig. 1 shows controls for creating new annotations while the top left shows a tree preview of the guideline model that is under construction. By clicking the *compile* button, this representation is translated into a metamodel-based flowchart model (step 2 from Fig. 3), which can be refined manually (step 3 from Fig. 3). Fig. 4 shows a screenshot of an example use of this editor. The left pane shows the editor palette, which enables the instantiation of the concepts from the syntax defintion. The middle pane shows an example flowchart diagram. In the screenshot, the "Hematoma" node from the upper left is selected and its details are shown in the rightmost editor pane. The pane enables a.o. associating reference papers (i.e., evidence) to the node. The editor instantiates models in such a format that they can be seemlessly processed by other special purpose MDE tools (e.g., for transformation and verification, see `http://www.eclipse.org/modeling/`).

We have also developed a prototypical code generator for realizing step 4 from Fig. 3. The complete tool-chain prototype was implemented by two junior programmers with basic Java programming skills. Students did receive guidance by one MDE expert, primarily in the use of the Epsilon framework [4]. Epsilon's Eugenia component has saved valuable time during the development of the CDS-specific flowchart model editor (shown in Fig. 4). The annotation editor (shown in Fig. 5) as well as its simple *"annotation to model compiler"* have been handcrafted.

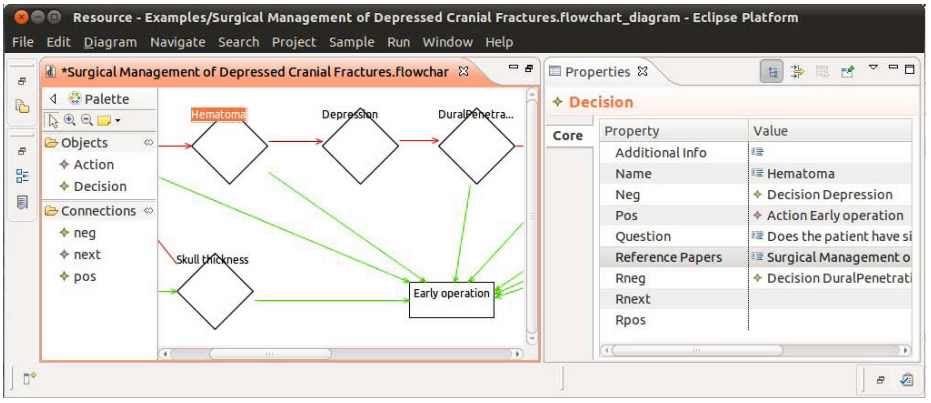


Fig. 4. Example Use of the MDE-based Clinical Guideline Editor Prototype

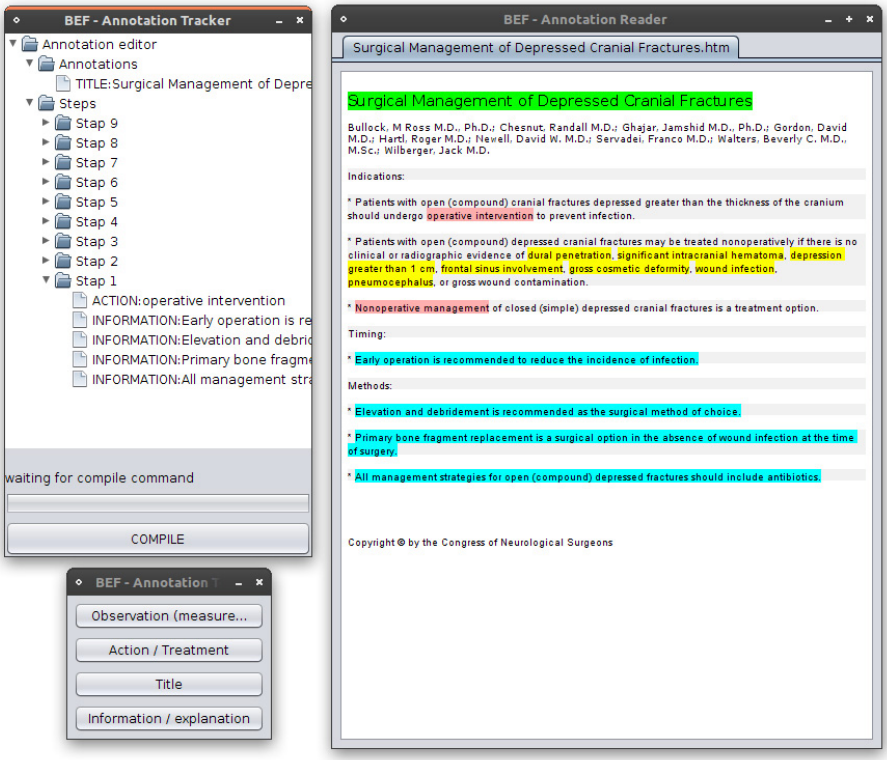


Fig. 5. Demonstration of support for elicitating model elements using text annotations

The need for custom model editors as well as annotation-based model extractors goes far beyond CGs and CDS, as discussed further in our related and future work section. Therefore, instead of replicating functionality in hand-crafted components for other metamodels, we investigated how the annotation and compilation support could be provided using modeling and transformation techniques.

3.2 Robust Support: Generating the Toolsuite from a Metamodel

Our aim is to provide MDE support across CDS and workflow applications for CGs, care pathways, reference pathways, etc. This section clarifies how we are tackling the lack of generic annotation-based model extraction tools by means of a small extension to the aforementioned Eugenia tool.

Eugenia is the MDE tool that we have used to transform the definition of the custom flowchart metamodel (the upper part of Fig 6) into a custom visual model editor. That existing Eugenia functionality is visualized by the bold arrow in the right part of Fig 6. The bold arrow in the left part of the figure represents the proposed new functionality for the Epsilon platform. That arrow visualizes how we intend to generate a custom (i.e., metamodel-specific) annotation tool from a metamodel definition. Coming back to the functionality of such a generated annotation tool: the left part of Fig. 6's *"Generated Annotation Editor"* contains a palette with buttons for creating specific annotations in the text that is shown in the right part of the box. From these annotations, the custom annotation editor (the generated tool at the bottom left of Fig. 6) would then create a model that could be further refined by the custom model editor (the generated tool at the bottom right of Fig. 6).

The following examples demonstrate the feasibility of implementing the proposed Eugenia extensions: the palette contains a button labeled *"Action/Treatment"* and a button labeled *"Title"*. The annotations for these buttons map directly to element attributes of the corresponding model. Therefore, it becomes possible to annotate the metamodel definition shown at the top of Fig. 6 in such a way that the buttons and their behavior is generated automatically by Eugenia. The two diagonal arcs on the figure illustrate which lines from the metamodel definition relate to which button in the annotation editor: for example, line 5 relates to the button labeled *"Title"*.

Note that a plain Eugenia metamodel definition (as shown at the top of Fig. 6) does not yet contain sufficient information to generate the metamodel-specific annotation editor. First of all, each line related to an annotation element would have to be supplemented with a label for the button (e.g., *"Title"* for line 5). Secondly, the line should be supplemented with a color for the text highlights (e.g., *green* for line 5 shown at the top of Fig. 6). Given these proposed extensions, line 5 could therefore be preceded by a line such as: "@annotation.element(button.name='Title',text.color='green')".

Further implementation details are outside the scope of this paper. We therefore leave it open whether the annotation editor and model editor shown at the bottom of Fig. 6 are separate tools or one integrated component. Regardless, generic model transformation languages/tools could be used to automatically

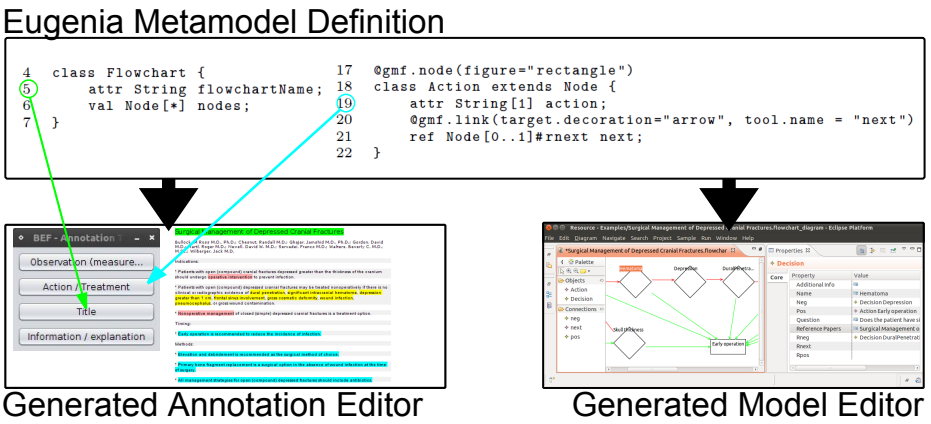


Fig. 6. Generating annotation tools from an extended Eugenia metamodel

produce and optimize output models. For the sake of persistence and consistency, we propose that the complete texts as well as the begin and end indices of annotations are stored also inside the metamodel-based output model.

4 Related and Future Work

We evaluated tool support for systematically deriving clinical guideline models from medical literature. We focussed on MDE tools since these are known to excel in the linkage of models with different purposes and at various levels of abstraction. MDE tools are also known to support the co-evolution of conceptual models with derived software systems. To the best of our knowledge, there are no experience reports on the linkage of MDE artifacts with unstructured documents.

Some isolated engineering efforts have been published outside the MDE context. For example, Lobach et al. [5] describe a conceptual process for translating informal guidelines into computer-interpretable representations. Concerning tool support, the Yale Center for Medical Informatics has developed and evaluated GEM Cutter II [6]. From the evaluation perspective, the Yale group has demonstrated that *an annotation-based approach to guideline model extraction is promising*, but also that *additional analysis is needed to determine the feasibility of offering "GEM-cut" recommendations nationally* (in the US). From the tooling point of view, we observe that GEM Cutter (1) is based directly on XML technologies rather than on formal metamodels, (2) does not provide support for visual models and (3) implicitly imposes one particular metamodel. We aim at tool support that excels on these three points: (1) we aim at an annotation-based model extraction infrastructure based on Eclipse ECore and EMF, which are the industry-standards for metamodeling in MDE; (2) we aim at supporting visual models, especially since medical papers often include flowchart based summaries; and (3) we aim at adaptable metamodels. We consider the last point of

particular relevance, since besides clinical guidelines there are various medical texts for which annotation-based model extraction is promising.

Terminology for candidate *models* is used rather confusingly both in medical and in information systems literature: different terms are used interchangeably, and the same term may have different meanings. Therefore, we have surveyed and classified the related literature. This has resulted in the following list of artifacts for which metamodels, editors, and extraction tools, need to be developed: Clinical Guidelines (CGs, e.g., [2]), Clinical Protocols (CPRs, e.g., [7]), Care Pathways (CPAs, e.g., [8]), Individual Care Pathways (ICPs, e.g., [9]), Assigned Pathways (APs) and Reference Pathways (RPAs). Our summarized definitions can be found in our previous work [1]. All these candidate models can be classified along two dimensions:

D1 (Patient Scope). The first dimension involves the scope of the description from the patient perspective: the most coarse grained descriptions aim at any type of patient, regardless of care groups. Other descriptions aim at multiple patients but within one specific care group. Finally, some descriptions are specific to an individual patient.

D2 (Provider Scope). In the provider aggregation dimension, some descriptions aim at multiple organizations while others are oriented at only one specific care organization.

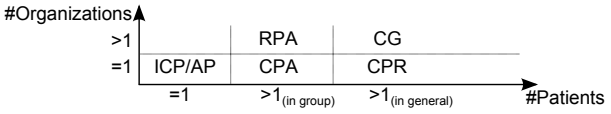


Fig. 7. 2D Classification of Process Oriented Care Descriptions

Fig. 7 shows that the classification of the aforementioned care descriptions in the proposed 2-dimensional space: when referring to the concrete goals and activities for one patient within one organization (=1, =1), one is considering ICPs and APs. When referring to the process descriptions for a group of patients within one organization ($> 1_{in\ group}$, =1), one is considering CPAs. When for such a group of patients one is referring to an abstract process description that is shared by multiple organizations ($> 1_{in\ group}$, >1) then one is considering RPAs. In the context of decision support for patients regardless of groups, CPRs are descriptions used within organizational boundaries ($> 1_{in\ general}$, =1) while CGs are used beyond these boundaries ($> 1_{in\ general}$, >1). All these artifacts are semantically related and since they can also evolve over time, our classification opens interesting opportunities for applying MDE techniques for co-evolution (e.g., those from Cichetti et al. [10]) in a challenging healthcare setting. In combination with our various metamodel-specific annotation tools, the traceability to related textual artifacts becomes manageable too. The practical integration of these techniques is the subject of our future work.

5 Conclusions

This short paper focuses on a specific MDE contribution: the development of a CDS based on the extraction of models from unstructured clinical guideline texts. However, our line of work is at the interface between Health Informatics and MDE research. Within the paradigm of using light-weight, custom editors instead of using heavyweight *"one size fits all"* editors (e.g., editors based on GEM, GLIF or SAGE), we have identified the lack of tools to extract purpose-specific models from unstructured texts. Besides presenting an exploratory, ad-hoc implementation of such a CDS-specific model extraction tool, we have discussed how a state-of-the-art MDE toolsuite can be extended for generating similar extraction tools in the large. Generating such tools is important since they are needed for a variety of healthcare and purpose-specific metamodels.

References

1. Van Gorp, P., Vanderfeesten, I., Dalinghaus, W., Mengerink, J., van der Sanden, B., Kubben, P.: MDE support for process-oriented health information systems: from theory to practice. In: Pre-proceedings of FHIES 2012 (August 2012)
2. Ross Bullock, M., et al.: Surgical management of depressed cranial fractures. Neurosurgery 58(suppl. 3), S56–S60, discussion Si–iv (2006)
3. Kubben, P.: Neuromind (December 2012), http://apps.digitalneurosurgeon.com/neuromind
4. Kolovos, D.S., Rose, L.M., Abid, S.B., Paige, R.F., Polack, F.A.C., Botterweck, G.: Taming EMF and GMF using model transformation. In: Petriu, D.C., Rouquette, N., Haugen, Ø. (eds.) MODELS 2010, Part I. LNCS, vol. 6394, pp. 211–225. Springer, Heidelberg (2010)
5. Lobach, D.F., Kerner, N.: A systematic process for converting text-based guidelines into a linear algorithm for electronic implementation. In: Proc. AMIA Symp., pp. 507–511 (2000)
6. Haskell, L.T., Monteforte, P.M.J., Shiffman, R.N., Coates, V.H., Nix, M.P.: S92 – applying the Guideline Elements Model (GEM) cutter II tool to guidelines represented in the national guideline clearinghouse (www.guideline.gov). Otolaryngol. Head Neck Surg. 143(suppl. 60-61) (July 2010)
7. Lorne Community Hospital: Anaphylaxis (2007), http://www.health.vic.gov.au/qum/downloads/anaphylaxis.pdf
8. South East Wales Cardiac Network: Integrated care pathway cardiac rehabilitation (May 2005), http://www.wales.nhs.uk/sitesplus/documents/986/ICPCardiacRehabPathwayJan2006.pdf
9. Elm Mount Units: Individual recovery/care plan review elm mount units (August 2012), http://www.mhcirl.ie/Inspectorate_of_Mental_Health_Services/ICPT/Elm%20Mount_CPT.pdf
10. Cicchetti, A., Di Ruscio, D., Eramo, R., Pierantonio, A.: Automating co-evolution in model-driven engineering. In: Proceedings of the 2008 12th International IEEE Enterprise Distributed Object Computing Conference, EDOC 2008, pp. 222–231. IEEE Computer Society, Washington, DC (2008)