

12. Validation- and Implementation-related issues

In this study, we concentrated on the knowledge representation aspects of guideline models. We did not cover execution-related issues, such as validation and implementation, which are very important for guideline usability. Because these issues are so important, we devote this section to describing, for each guideline model, validation and implementation aspects that were not studied by comparing the different guideline models.

a. Asbru

The validation of Asbru plans is carried out on a series of levels:

1. Syntactic correctness is proven by the XML-editor, based on the Asbru-DTD
2. Semantic issues, such as the mapping of references in different name spaces, are covered by a compiler that was generated by Pontifex [9]. Pontifex takes a language description in HSL (Harmless Specification Language) and produces a parser, classes representing the elements, a DTD, and documentation
3. The intelligent editor PIXEE performs checks for semantic issues such as matching definition and use of variables and parameters. Its main aim is to improve the overview of the various aspects of a plan library, providing intelligent folding of language elements.
4. A special verification module detects sets of anomalies that hint at possibly contradicting definitions in the plan library [3]. Currently this is done by hand. An implementation is in progress.
5. There will be a simulation mode to observe the plans' reactions to input that is directly entered or pre-recorded. This will allow to test a plan library
6. AsbruView is used to structure the overall topology of the plan library. The domain-dependent part of the plan library is separated from the independent part and several domains can coexist in a plan library. This allows for different definitions of parameters, patient data records, and primitive plans for different domains. Of course, these definitions must be compatible on the top level accessed by the plans
7. Resource management is not the focus of Asbru, although some aspects like specification of necessary resources are covered.

8. Events concerning the execution of plans are modeled as changes of parameter values.

b. GLIF

GLIF includes versioning and validity attributes: Author, encoder, GLIF version, title, guideline_version, date, developing_institution, and adapting_institution.

There is GEL parser that can test logical validity of expressions.

Protégé has minimum / maximum cardinality constraints etc, lower / upper limits on numerical values. In addition, authoring in Protégé does type checking. Since Protégé can be used to author GLIF guidelines, Protégé carries out these checks.

Following is a list of manual checks that can be done to catch safety bugs:

1. Checking that branch and synchronization steps are balanced
2. Checking that all case step options are mutually exclusive
3. Checking that decision options cover the entire range
4. Checking that data items referred to by logical expressions are all defined
5. Checking that each guideline has all the necessary entry points (patient-state steps marked as entry points)
6. Checking that all the steps that do not have outgoing arcs represent exit points

c. PRODIGY

PRODIGY has tools that enable multiple authors to work on the same guideline and check each other's work. Medically-oriented people review the guideline models. The PRODIGY3 team implemented three complex chronic disease management guidelines (asthma, hypertension, angina) and 150-160 simple guidelines in two vendor systems. PRODIGY1 is supported by all vendor systems in the UK. The team is subject to external review.

d. PROforma

PROforma has various 'wrapping' software enable database access and web-delivery.

Arezzo provides a set of syntax directed checking functions (for task and data incompleteness, contradictions, task execution faults). The syntax checks are

integrated into the authoring environment through the warning and error tags on the guideline overview display.

Although Arezzo can display the *PROforma* representation of a guideline, it does not use it as an interchange format. The Newcode implementation does export in *PROforma* syntax.

Both Arezzo and the Newcode implementations of *PROforma* provide an execution engine and effective test environment, Arezzo in particular provides a very wide range of test functions.

e. EON

The EON team has implemented the Athena hypertension guideline in three geographical areas that span several clinics of the US Veteran's Administration. Athena has been already running over a year in the Palo Alto clinic. Athena gets dumps of data from the EMR. The dumps are mapped to EON's patient data model overnight. This is not characteristic of EON but of this particular implementation. A customizable pop-up window appears for patients who have hypertension when the physician needs to examine them.

As is true for GLIF, EON uses Protégé as its authoring tool. Therefore, minimum / maximum cardinality constraints, lower / upper limits on numerical values can be checked. Type checking is also supported. PROTÉGÉ has a formal first order logic expression language that guideline authors can use to write integrity constraints that are then enforced by Protégé.

The guideline-execution engine can be event-driven. The implementation environment depends on external implementation systems that can listen to event and throw exceptions. EON and PRODIGY do not model event triggering of fine-grained actions. However, particular installations of EON systems at a clinic may use an event to invoke the decision-support system.

f. GUIDE

The guideline software integrates the clinical guideline model with workflow management. In addition to the medical knowledge that is represented by the GUIDE

model, organizational knowledge is represented in an organization model. The organization ontology represents entities such as organization, organization unit, organization goal, agent, role, resource, as well as relationships among those entities. Each organization that implements a guideline creates instances of the organization model entities.

The workflow management tools can handle situations in which guideline users do not comply with the guideline's recommended tasks. The tools offer the user alternative tasks and let her enter a comment about the motivation for not complying with the guideline. The reasons for non-compliance can also be inferred a posteriori by applying rules that exploit ontological relationships among tasks and knowledge about the technological and human resources that are needed to accomplish tasks. For example, a rule might be "If task_i requires at least two roles (i.e., human agents that play a certain role in the organization) to carry it out and these roles are present in the organization then the cause of non-compliance is likely due to lack of internal coordination.

GUIDE has an expression language parser that produces SQL queries to retrieve data from the EMR and use it for deriving recommended tasks, according to the rules that have been translated from GUIDE into the workflow model.