TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

DIPLOMARBEIT

Structured Knowledge Acquisition for Asbru

ausgeführt am Institut für Softwaretechnik
der Technischen Universität Wien

unter der Anleitung von
ao.Univ.Prof. Dr. **Silvia Miksch**
und
Dr. **Robert Kosara**

durch

Peter Votruba
Fahrbachgasse 12/2/20
1210 Wien
Matr.Nr.: 9625117

Kurzfassung

Es gibt verschiedenste Möglichkeiten, medizinische Behandlungsrichtlinien so aufzubereiten, daß sie von einem Programm verarbeitet werden können. Alle bisherigen Ansätze haben jedoch den Nachteil, daß keine direkte Verbindung zwischen dem Original und der digitalisierten Version hergestellt werden kann. Es ist daher oft schwierig, im Nachhinein den Umwandlungsprozeß nachzuvollziehen.

Diese Arbeit stellt ein Programm vor, das speziell entwickelt wurde, um diesen Nachteil auszugleichen. Es entstand als Teil des *Asgaard*-Projekts, welches die Entwicklung von Methoden und Programmen für die Erstellung, Überprüfung und Ausführung von medizinischen Behandlungsplänen zum Ziel hat. Das sogenannte *Guideline Markup Tool (GMT)* erleichtert die Übersetzung von Behandlungsplänen in die Modellierungssprache *Asbru*, die im *Asgaard*-Projekt verwendet wird. Dies wird einerseits erreicht durch die Möglichkeit, Verknüpfungen zwischen dem ursprünglichen Behandlungsplan und der *Asbru*-Repräsentation zu erstellen. Andererseits können mit dem GMT *Asbru*-Dokumente mit Hilfe von Makros, die jeweils mehrere *Asbru*-Elemente beinhalten, einfach zusammengebaut werden.

Im Rahmen dieser Diplomarbeit wurde außerdem ein Makrofile für das GMT erstellt, welches bereits viele nützliche Makros für die Erstellung von *Asbru*-Dokumenten zur Verfügung stellt.

Um die Verwendbarkeit des GMT und die Nützlichkeit der *Asbru*-Makros nachzuweisen, wurde eine Evaluierung mit acht Teilnehmern mit jeweils unterschiedlichen Fähigkeiten durchgeführt.

Abstract

There are several efforts to map clinical guidelines to a form, which can be further processed by special programs. All these approaches have a common drawback: they do not allow to define a persistent connection between the original guideline and its computerized version. Therefore, it is often difficult to comprehend the translation process at a later time.

This thesis introduces a tool that was designed to overcome this drawback. It was developed as part of the *Asgaard* project, which aims at developing methods and tools for authoring, validation and execution of clinical guidelines. The so-called *Guideline Markup Tool (GMT)* facilitates the translation of guidelines into the guideline representation language *Asbru,* which is also part of the *Asgaard* project. On the one hand, this tool enables creating links between the original guideline and its *Asbru* representation. On the other hand, it supports constructing a new *Asbru* guideline through providing macros, which combine one or more *Asbru* elements with their attributes.

As a second part of this work, a macros file for the GMT has been written, which provides many useful macros for the creation of *Asbru* guidelines.

To examine the usability of the GMT and the usefulness of the *Asbru* macros, an evaluation with eight participants with different skills has been performed.

Table of Contents

List of Figures

Acknowledgments

I wish to thank a number of persons, who enabled this work:

First of all, I would like to extend my sincere thanks to my supervisor *Silvia Miksch* for her great support.

I wish to thank *Robert Kosara*, who volunteered as co-supervisor besides his actual work.

Very important for the development of the Guideline Markup Tool was the first user, *Marije Geldof*. She already used the GMT for her thesis, long before I have finished working on it. Besides sending me many suggestions for improvements, she pointed out some nasty bugs.

Another person who helped me in my work was *Mar Marcos*. Due to her experience in creating Asbru guidelines, she gave me some invaluable advices for the development of the macros file.

Finally, the evaluation participants should not be forgotten: *Katharina Kaiser*, *Georg Duftschmied*, *Andreas Seyfang*, *Christian Popow*, *Monika Lanzenberger*, *Wolfgang Aigner*, *Peter Messner* and *Klaus Hammermüller*.

Additionally, I wish to thank my family for their personal support and my girlfriend for always being there for me.

This thesis was supported by "Austrian Science Fund", grant 12797-INF.

1 Introduction

This chapter first introduces the application domain and exposes the motivation of this work. Additionally, it gives an overview of the structure of this thesis.

1.1 Motivation

Clinical protocols and guidelines have been introduced to standardize treatment methods of physicians. They are intended to define each step of a treatment for specific diseases to reduce proneness to errors. Conventional guidelines are written as plain documents, sometimes including tables or flow charts for better illustration of important facts. Often these documents contain misleading ambiguities, which reduce the attraction of guidelines.

Several approaches have been carried out with the aim of improving the usefulness of guidelines, by trying to model them in a machine-readable form using a guideline modelling language. In most cases, the main goal is to systematically validate guidelines. Additionally, software systems have been developed, which support the computerization and application of guidelines.

The *Asgaard*[1] project ([Shahar et al., 1998]; see [URL #1]) was initiated to develop task-specific problem-solving methods that perform design, execution, and validation of clinical guidelines. Within the *Asgaard* project, a guideline representation language, called *Asbru*[1], has been developed. Additionally, *Asbru*View, a simple knowledge acquisition and visualization tool, has been created. It is well suited for clinical professionals through its heavy use of graphical metaphors.

Other projects either use their own tools for translating guidelines or rely on general knowledge acquisition tools, like *Protégé* (see [URL #35]). All existing guideline tools have their inherent advantages and disadvantages, but they have all one common drawback. They do not provide any possibility to establish connections between the original guideline and its computerized version. For example, if the validation of an *Asbru* document leads to contradictions, it would be helpful to see the corresponding parts of each inconsistent *Asbru* element in the original guideline. A new guideline tool has been demanded, which provides such a feature.

This thesis introduces the *Guideline Markup Tool (GMT)*, a new tool supporting the translation of guidelines into their *Asbru* representation. It was specially designed to overcome the above-mentioned drawback of other tools. It enables the creation of links between the original guideline and its *Asbru* representation. Furthermore, it facilitates the assembling of an *Asbru* guideline, by providing macros, each combining several *Asbru* elements.

Additionally, this thesis presents a macros file, which contains many useful macros for constructing *Asbru* guidelines.

[1] Many of the names in the Asgaard project are based on names from Norse mythology. *Asbru* (also known as *Bifrost*), for example, is the rainbow bridge that leads to *Asgaard*, the home of the Norse gods.

1.2 Structure of the Thesis

This document is structured into four parts. The first part includes chapters containing necessary background information. The second part is about the actual work I have done. The subsequent part presents the evaluation of my work and the conclusions of this thesis. Finally, there is an appendix providing supplementary material like the questionnaires used during the evaluation of the GMT.

The three main parts are formed by the following ten chapters: The first chapter of the first part contains definitions of basic terms. The next chapter provides an overview of three structured knowledge acquisition and modelling techniques. The third chapter introduces *Asbru* and gives an overview of several other guideline modelling methods. The next chapter presents some guideline technologies. A summary of the presented approaches rounds out the first part.

The *Guideline Markup Tool* will be described in chapter 6. In the second chapter of this part, the macros file will be explained.

The evaluation of the GMT will be presented together with the results in chapter 8. The ninth chapter provides a summary of this thesis. A list of possible improvements of the GMT and the macros file concludes this part.

There are two different lists of references at the end of this document. Papers and books are listed in the *Bibliography* (p. 62) and referenced using the author's name(s) and the year of publication, for example [Geldof, 2002]. On the other hand, websites are listed in *Related Websites* (p. 66) and referenced using a sequential number, like [URL #1].

I Problem Analysis

2 Definitions of basic Terms

This chapter defines several basic terms, which are essential for the understanding of the problem analysis part of this thesis.

- Clinical Practice Guidelines and Clinical Protocols

  "*Clinical guidelines are a set of schematic plans for management of patients who have a particular clinical condition*" ([Miksch et al., 1997a]) to support clinical practitioners in their decision-making. In the glossary of the *OpenClinical* website ([URL #2]), the term "clinical practice guideline" is defined as "*validated policy statements representing best clinical practice, used to support standardised patient care*".

  "*Clinical protocols are a more detailed version of clinical guidelines, often used when guidelines need to be applied uniformly to enable statistical analysis of the outcomes*" ([Miksch et al., 1997a]). [URL #2] defines the term "clinical protocol" as "*a standard set of tasks that define precisely how classes of patients should be managed or treated*".

  To stay consistent, only the terms *clinical guideline* or *guideline* will be used throughout this thesis.

- Declarative Knowledge *versus* Procedural Knowledge

  Declarative knowledge represents awareness of particular objects, events or ideas. This type of knowledge can be paraphrased as *knowing that*. When a person *knows that*, she is able to describe the objects of that knowing, but that means not necessarily that he/she is able to use that knowledge, since declarative knowledge does not imply understanding ([Jonassen et al., 1993]).

  Procedural knowledge, on the other hand, defines how declarative knowledge is being used. This type of knowledge can be paraphrased as *knowing how*. Solving problems, forming plans and making arguments are examples of activities that lead to procedural knowledge. Declarative knowledge provides the conceptual basis for procedural knowledge ([Jonassen et al., 1993]).

  In other words, *declarative knowledge* contains facts about objects and *procedural knowledge* tells how to apply these facts.

- Ontology

  The term "ontology" comes originally from philosophy, where it stands for "*the branch of metaphysics that deals with the nature of being*" (definition taken from *The Collins English Dictionary*).

In the domain of knowledge engineering, it has a different sense: The *OpenClinical* glossary ([URL #2]) defines the term as "*a set of concepts, their attributes and the relationships between them within a given application area, typically represented as a knowledge base*". Another definition comes from [Gruber, 1992]: "*An ontology is an explicit specification of a conceptualisation*", whereby "*conceptualisation is an abstract, simplified view of the world that we wish to represent for some purpose*".

The main application area of ontologies in knowledge engineering is knowledge sharing – regarding the medical domain, special ontologies have been developed to enable dissemination of clinical guidelines.

➢ <u>Schema Language *versus* Ontology Language</u>

A schema language is a set of grammatical rules for describing schemas in general. The term "schema" is defined by the *SCHEMAS – Glossary* ([URL #3]) as "*a set of metadata elements representing the attributes of a resource*", whereas "*each element will have a name and associated semantics*". Some examples for schema languages regarding XML are DTDs, W3C Schemas and RELAX NG.

An ontology language can be much more complex than a schema language. In simple terms, an ontology language is used to describe or specify an ontology. It usually introduces concepts (classes, entities), properties of concepts, relationships between concepts, and additional constraints.

Examples for ontology languages are KIF (<u>K</u>nowledge <u>I</u>nterchange <u>F</u>ormat), DAML+OIL, RDF Schema, Ontology Markup Language (OML) and Operational Conceptual Modelling Language (OCML). A good overview of these languages and others can be found in [Corcho & Gómez, 2000] and [Bechhofer, 2002].

3 Structured Knowledge Acquisition and Modelling

Asbru is a modelling language for clinical guidelines and protocols, which can be seen as a special kind of structured knowledge. The Guideline Markup Tool, which was developed especially for *Asbru*, is, on the other hand, a special kind of a knowledge acquisition tool that heavily makes use of structured knowledge in the form of the macros file. Therefore, it would be helpful to make a side trip to the basics of structured knowledge techniques.

This chapter defines the term "structured knowledge" and presents the state-of-the-art of structured knowledge techniques.

3.1 Definition of Structured Knowledge

For a quite long time, knowledge has been classified into two types, declarative knowledge and procedural knowledge.

[Jonassen et al., 1993] propose an intermediate type of knowledge called *structured knowledge*. It facilitates the translation of declarative into procedural knowledge and supports the utilization of procedural knowledge. Structured knowledge describes how concepts within a domain are interrelated. It is not enough to *know that*. In order to *know how*, you must *know why*. Structural knowledge provides the conceptual basis for *why* – it describes *how* the declarative knowledge is interconnected.

3.2 State-of-the-Art

Three different projects have been selected to present the state-of-the-art in the wide area of computer-based structured knowledge technologies. CommonKADS is a methodology designed for the development of knowledge based systems (KBS). DAML+OIL is a semantic markup language for Web resources. Finally, *Lixto* is a technology, which translates the relevant parts of web pages into XML.

3.2.1 Common KADS

CommonKADS ([URL #6]) is a methodology to support structured knowledge engineering. It has been developed in the context of the European ESPRIT IT Programme and validated by many companies and universities. Nowadays it is the European de facto standard for knowledge analysis and knowledge-intensive system development.

"*The aim of CommonKADS is to fill the need for a structured methodology for knowledge based system (KBS) projects by constructing a set of engineering models built with the organization and the application in mind*" [Schreiber et al., 1994].

CommonKADS consists of several diagram-based models which reflect the different aspects of knowledge: ([Vollebregt et al., 1999], p. 6/7)

- ***Organisation Model****:* concerned with the organisation in which the KBS must function
- ***Task Model***: modelling inputs, outputs, preconditions and performance criteria of the global task
- ***Knowledge Model****:* a detailed but implementation-independent description of the knowledge used to perform the desired tasks, and the role that different knowledge components play in problem-solving
- ***Agent Model****:* assigning tasks to various agents such as humans, computers, KBS, etc.
- ***Communication Model****:* stating the communication between the various agents
- ***Design Model****:* describing the technical specification of the system

The central model is the *knowledge model*. The knowledge in this model is separated into three categories (also called "layers"):

- ***Domain Knowledge*** describes object properties, relations and concepts in the application domain.
- ***Inference Knowledge*** describes the reasoning steps (or inference actions) that can be performed using the *domain knowledge*.
- ***Task Knowledge*** specifies the execution order of the inference steps.

The coherences between the three layers in the knowledge model can be seen in Figure 3.1 (taken from [Vollebregt et al., 1999]). Primitive tasks at the task layer are mapped to inference steps at the inference layer, and knowledge-roles at the inference layer are mapped to knowledge at the domain layer.

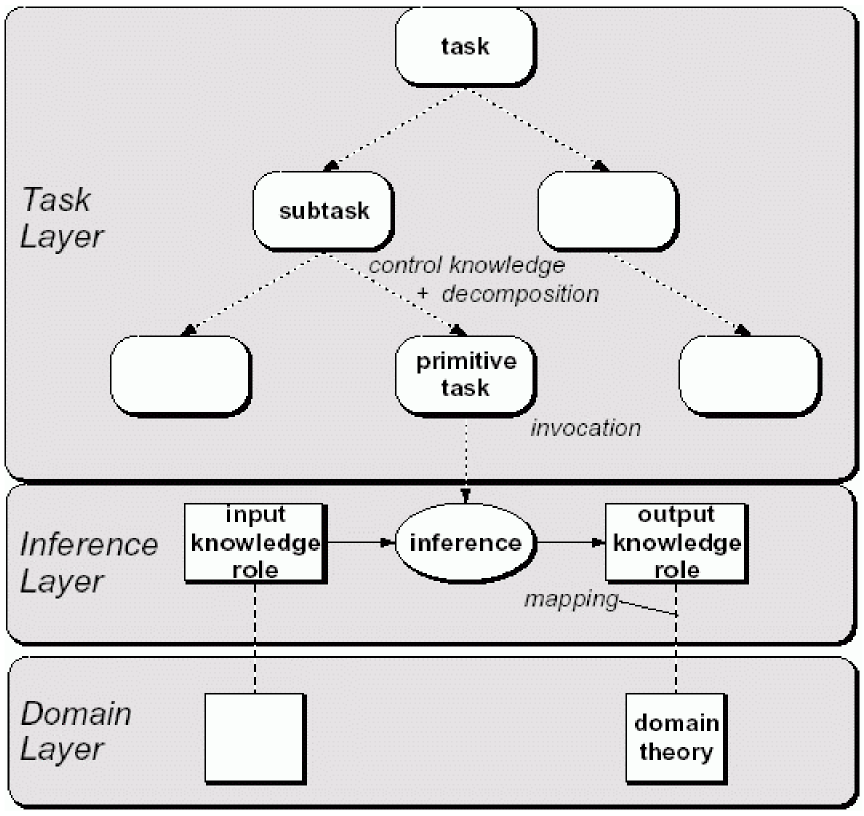


Figure 3.1: Structure of the *CommonKADS* Knowledge Model

3.2.2 DAML+OIL

RDF (Resource Description Framework) is an infrastructure that enables the encoding, exchange and reuse of structured metadata. It was developed by the W3C[1] at about the same time as XML. RDF has an explicit model for expressing object semantics using XML. The RDF model has proven to be successful because of its simplicity. As an enhancement of RDF, the W3C developed a schema language, called RDF Schema (RDFS), to provide basic structures such as classes and properties.

In response to W3C's RDF the US *Defense Advanced Research Projects Agency* (DARPA) started the DAML (DARPA Agent Markup Language) project [URL #11], which released DAML-ONT, a simple language for expressing more sophisticated RDF class definitions than permitted by RDFS. The DAML project combined efforts with the *Ontology Inference Layer* (OIL) project [URL #12], another ontology representation language, which uses constructs from frame-based languages. The result is DAML+OIL, a semantic markup language for Web resources with clean and well-defined semantics. It is not just a schema language but also an ontology language, providing primitives that support the general representation of knowledge. The most recent release is "DAML+OIL (March 2001)" (see [URL #8] and [URL #9]), which now provides values from XML Schema datatypes.

DAML+OIL, is written in RDF, whereas RDF, in turn, is written in XML, using XML Namespaces. This means that DAML+OIL markup is a specific kind of RDF markup and thus a specific kind of XML markup.

[1] The World Wide Web Consortium (W3C) was founded to enhance the interoperability of the World Wide Web. It is responsible for important web standards, like HTML, CSS, XML and XSL [URL #7].

The example in Figure 3.2 (borrowed from [URL #10]) shows a part of a DAML+OIL document. It defines a basic type "Animal" with two subclasses, "Male" and "Female", which are disjoint (since an animal cannot be both male and female).

```
<daml:Class rdf:ID="Animal">
  <rdfs:label>Animal</rdfs:label>
  <rdfs:comment>
    This class of animals is illustrative of a number of ontological idioms.
  </rdfs:comment>
</daml:Class>
<daml:Class rdf:ID="Male">
  <rdfs:subClassOf rdf:resource="#Animal"/>
</daml:Class>
<daml:Class rdf:ID="Female">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <daml:disjointWith rdf:resource="#Male"/>
</daml:Class>
```

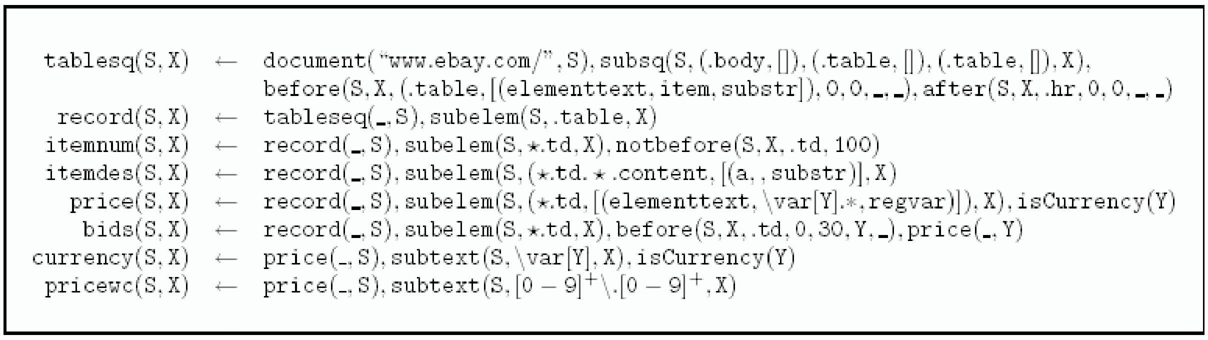
Figure 3.2: DAML+OIL Example

3.2.3 Lixto

Lixto ([Toprakkiran, 2001]; see also [URL #14] and [URL #15]) is a system for generating wrapper programs, which are used to extract desired information from HTML pages on the World Wide Web and to translate them into XML. The generation task takes place semi–automatically, with interaction between the designer and the system. This interaction happens through a simple visual user interface, the *Lixto Visual Wrapper*. The user creates programs for example pages chosen by him in a visual manner, making use of the new methods of *Lixto* for identifying and extracting relevant parts of the pages. The constructed programs are internally maintained in a new declarative and logic–based language, called *Elog*. At the end of the generation task, the wrapper programs can be saved and afterwards automatically applied to the class of similarly structured pages.

Lixto guarantees a continually updated XML–translation for changing HTML pages. In addition, *Lixto* is easy and fast to learn, even for non–technical users, because the user interface is extremely simple and because the users neither have to deal with the internal details of *Elog* nor with any HTML code of the web pages.

On the other hand, more advanced users may use *Elog* to refine visually constructed programs. Figure 3.3 (taken from [Baumgartner et al., 2001]) shows an example of an *Elog* program with some typical Prolog-like extraction rules.



```
  tablesq(S,X)  ←  document("www.ebay.com/",S), subsq(S,(.body,[]),(.table,[]),(.table,[]),X),
                   before(S,X,(.table,[(elementtext,item,substr]),0,0,_,_), after(S,X,.hr,0,0,_,_)
   record(S,X)  ←  tableseq(_,S), subelem(S,.table,X)
  itemnum(S,X)  ←  record(_,S), subelem(S,⋆.td,X), notbefore(S,X,.td,100)
  itemdes(S,X)  ←  record(_,S), subelem(S,(⋆.td.⋆.content,[(a,,substr)],X)
    price(S,X)  ←  record(_,S), subelem(S,(⋆.td,[(elementtext,\var[Y].∗,regvar)]),X), isCurrency(Y)
     bids(S,X)  ←  record(_,S), subelem(S,⋆.td,X), before(S,X,.td,0,30,Y,_), price(_,Y)
 currency(S,X)  ←  price(_,S), subtext(S,\var[Y],X), isCurrency(Y)
  pricewc(S,X)  ←  price(_,S), subtext(S,[0−9]+\.[0−9]+,X)
```

Figure 3.3: *Elog* Program for Extraction of Information from the *eBay* Website

4 Guideline Modelling Methods

This chapter first introduces the clinical guideline representation language *Asbru*. Then it presents a couple of other guideline modelling methods.

4.1 Introduction to Asbru

Asbru ([Miksch et al., 1997a], [Miksch et al., 1997b], [Roomans, 2001]) is a time-oriented, intention-based, skeletal plan-specification language that is used to represent clinical protocols. The Asbru language was developed as a part of the *Asgaard* project, to model clinical guidelines and protocols.

Asbru is called time-oriented, because of its heavy use of *time annotations*. A time annotation in Asbru allows representation of uncertainty in the starting time, ending time and duration of an event, through the definition of an earliest starting time and a latest starting time, respectively an earliest ending time and a latest ending time and a minimum and maximum duration.

The term "intention-based", with respect to *Asbru*, means that the goals (or intentions) of a guideline are defined as an essential part of that guideline. For example, such intention could be "*avoid more than two episodes of hyperglycaemia per week*".

Guidelines are modelled as a set of hierarchical skeletal plans. The term "skeletal plans" denotes plan schemata at various levels of detail, which capture the essence of the procedure, but leave room for execution-time flexibility in achieving goals [Friedland & Iwasaki, 1985].

The **plan** is the basic building block of an Asbru guideline – a guideline consists of several plans and a plan, on the other hand, may contain other *sub*-plans.

A plan may consist of the following five major components:

- **Preferences** are plan parameters that express a kind of behaviour of the plan or constrain the selection of a plan to achieve a given goal.
- **Intentions** are high-level goals at various levels of the plan. They can be thought as temporal patterns to be maintained, achieved, or avoided.
- **Conditions** are temporal patterns that need to hold at particular plan steps to cause a specific state transition of the plan instance. By using conditions, one can decide which plan should be applied due to a certain state of the patient that the plan was designed for.
- **Effects** represent the overall effect of a plan on parameters when applied to a patient and describe the relationship between plan arguments and measurable parameters.
- **Plan Body** may contain different steps, primarily user-performed actions, sub-plans and pointers to other plans (which enables reusing of plans).

Plans can be divided in four types, depending on the temporal order of their steps:

- **sequential**: All steps of a plan are executed in a strict order.
- **any-order**: All steps are executed in a sequence, but without certain order.
- **parallel**: All steps are executed concurrently.
- **unordered**: Each step may start whenever appropriate, without any order.

Each plan has an attribute that defines if the plan has to wait for all sub-plans to complete successfully or not. Furthermore, there are *cyclical* plans, which will be automatically retried, if they have been completed unsuccessfully.

This chapter is not intended to provide a full explanation of *Asbru*. As mentioned above, there exist several papers which focus on Asbru. For information about all *Asbru* elements, see the "Asbru Reference Manual" ([Seyfang et al., 2002]).

Since a document type definition (DTD) for Asbru has been written, Asbru guidelines can be authored in XML. This leads to the improvement of *Asbru*View (chapter 5.1, p. 12), and to the development of PIXEE (chapter 5.2, p. 14) and finally GMT (chapter 7, p. 21).

4.2 Overview of other approaches

Besides *Asbru*, there exist of course plenty of other projects, which focus on guideline modelling. This chapter gives a short overview of these approaches and tries to state the main points of each of them. See chapter 6.1 (p. 20), for a comparison of the various guideline modelling methods.

4.2.1 EON

The **EON** architecture ([Musen et al., 1996]; [URL #18]), developed at the *Stanford Medical Informatics group*, consists of a suite of models and software components for the creation of guideline-based applications. Its guideline model (called *Dharma* model) defines guideline knowledge structures such as selection criteria, abstraction definitions, guideline algorithm, decision models and recommended actions. Conditional goals are associated with guidelines and sub-guidelines. The guideline algorithm is represented as a set of scenarios, action steps, decisions, branches and synchronization nodes. [URL #19]

Encoding of EON guidelines is done with the widely used *Protégé-2000* tool [URL #35], a general ontology and a knowledge-base editor.

4.2.2 GEM

GEM ([Shiffman et al., 2000]; see [URL #20]) was developed by the *Yale Center for Medical Informatics* to organize guideline knowledge into a standardized structure. Using this model, information contained in a wide variety of guidelines can be organized into a hierarchical collection of elements, thus it is called the Guideline Elements Model.

A strength of this model is that it encodes considerable information about guideline recommendations in addition to the recommendations themselves, including the reason for each recommendation, the quality of evidence that supports it, and the recommendation strength assigned by the developers. [URL #21]

Like Asbru, the GEM Format is based on XML. A special tool, called GEM-Cutter (chapter 5.3, p. 15), which facilitates simple conversion from textual guidelines into GEM XML files, was developed within the GEM project.

4.2.3 GLARE

GLARE (GuideLine Acquisition, Representation and Execution) is a domain-independent system for acquiring, representing and executing clinical guidelines. GLARE has been under development since 1997 by the *Dipartimento di Informatica*, *Università del Piemonte Orientale*, Alessandria, Italy, in co-operation with the *Laboratorio di Informatica Clinica, Azienda Ospedaliera San Giovanni Battista*, Torino, Italy.

The GLARE representation language is designed to achieve a balance between expressiveness and complexity. The formalism consists of a limited, but focused and clearly understandable set of primitives. It is made up of different types of actions: plans (i.e. composite actions, hierarchically decomposable in their sub-actions) and atomic actions. Atomic actions can be queries, decisions, work actions and conclusions. All actions are linked by control relations (e.g. sequence, alternative, repetition), defining their order of execution.

The system includes an acquisition tool and an execution tool. [URL #22]

4.2.4 GLIF

GLIF ([Ohno-Machado et al. 1998]; see [URL #23]) is being developed by the *InterMed Collaboratory,* which includes *Stanford Medical Informatics, Harvard University*, *McGill University* and *Columbia University*.

The main goal of GLIF (GuideLine Interchange Format) is to develop a standardized approach that will facilitate sharing of computer-based clinical practice guidelines. The GLIF specification consists of an object-oriented model and a corresponding text syntax. GLIF contains a set of classes to represent the content of a guideline, each with specific data types and attributes. Collections of steps are linked together to define the flow of control of a guideline. [URL #24]

GLIF guideline representations are exchanged in RDF format, to facilitate an open process. An RDF Schema defines the metadata definitions of GLIF's object model. Guideline instances that conform to the RDF Schema are specified in RDF markup. The RDF instance files and the RDF GLIF Schema are easily interchanged among institutions that use GLIF as the language for representing encoded guidelines.

4.2.5 GUIDE

GUIDE is part of a guideline modelling and execution framework developed at the University of Pavia. The GUIDE method is based on *Petri Nets*[1]. The strength of *Petri Nets,* when applied to healthcare, is their ability to support the modelling of complex concurrent processes and integrate clinical tasks specified in guidelines with organisational models to manage patient care workflow. Petri Nets have been extended to support improved modelling of time, data and hierarchies. GUIDE uses the underlying Petri Net formalism to be able to support the representation of sequential, parallel and iterative logic flows [URL #25].

4.2.6 PRODIGY

PRODIGY (Prescribing RatiOnally with Decision-support In General-practice studY) is an ongoing research and development project to develop and test the concept of computerised decision support systems aiding general practitioners prescribing in the UK ([URL #26]). The research team is based at the *Sowerby Centre for Health Informatics* at Newcastle (SCHIN).

[1] … a rigorous formalism for modelling concurrent processes, invented by Carl Adam Petri in the 1960s.

The PRODIGY model was created to model guidelines for the management of chronic diseases, such as asthma, hypertension and angina, in primary care. Essentially, it supports modelling series of decisions that a general practitioner may have to make in different patient encounters. The model enables a guideline to be organized as a network of patient *scenarios*, management decisions and action steps, which produce further scenarios, whereas scenarios are patient states defined by the patient's condition and current treatment. [URL #27]

4.2.7 PRO*forma*

The **PRO*forma*** ([Vollebregt et al., 1999], [Bury et al., 2000]; [URL #28]) architecture has been developed at *Cancer Research UK*. It contains a formal knowledge representation language designed to capture the content and structure of a clinical guideline in a form that can be interpreted by a computer.

PRO*forma* supports the definition of clinical guidelines and protocols in terms of:

- a well-defined set of tasks that can be composed into networks representing plans or procedures carried out over time.
- logical constructs which allow the details of each task and inter-relationships between tasks to be defined using templates.

There are four classes of PRO*forma* tasks (see Figure 4.1):

- **plan**: a sequence of several sub-tasks or components. Plan components usually have an ordering to specify temporal, logical or source constraints.
- **decision** is used where one of the candidates for the result of the decision must be chosen from a given set using arguments pro and contra.
- **action**: a procedure that has to be executed outside of the computer system, like an injection.
- **enquiry**: requests information needed to execute a certain procedure.

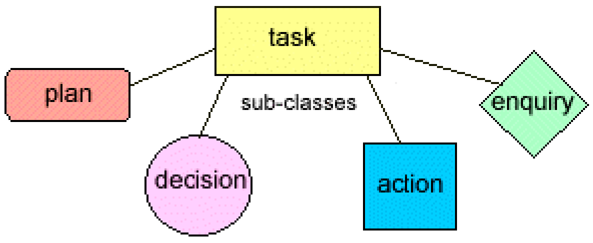


Figure 4.1: PRO*forma* task hierarchy

The properties and behaviour of each task type are determined by its attributes and the values attached to them. These determine how and when a task is processed and when the electronic guideline, of which they form a part, is executed.

There are currently two authoring tools, that have been especially created for the PRO*forma* language (see chapter 5.4, p. 17).

5 Guideline Acquisition Methods

The last chapter introduced several guideline modelling methods. This chapter presents some guideline acquisition technologies. Not all of the projects presented in chapter 4 have released own guideline tools. Besides the Asgaard/Asbru project, only GEM and PRO*forma* are represented with their tools. In addition to those tools, the Guide-X methodology will be introduced.

5.1 *Asbru*View

***Asbru*View** ([Miksch et al., 1998], [Kosara, 1999], [Kosara & Miksch, 2001]) was developed within the Asgaard project to facilitate the creation, editing and visualization of *Asbru* (see chapter 4.1, p. 8) files.

When the development of *Asbru*View started, Asbru was defined in a LISP-based syntax, which was very hard to understand for medical experts and therefore unusable for them. Thus, the main reason for the development of *Asbru*View was to build a tool, which could be easily used by every clinician without any difficulty. This has been accomplished through the usage of graphical metaphors.

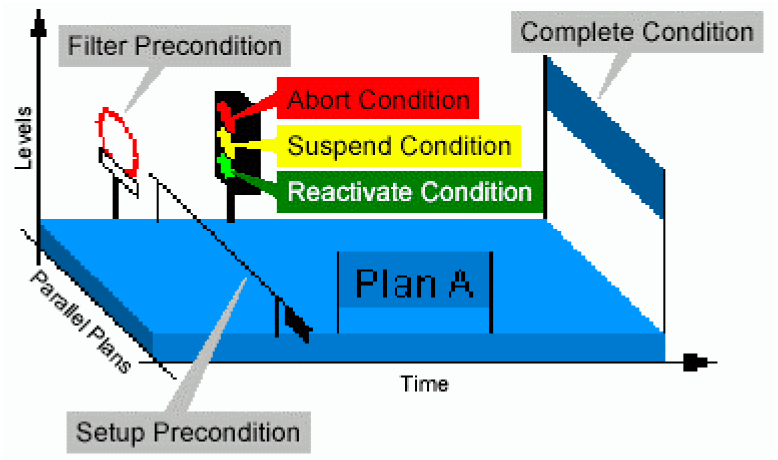
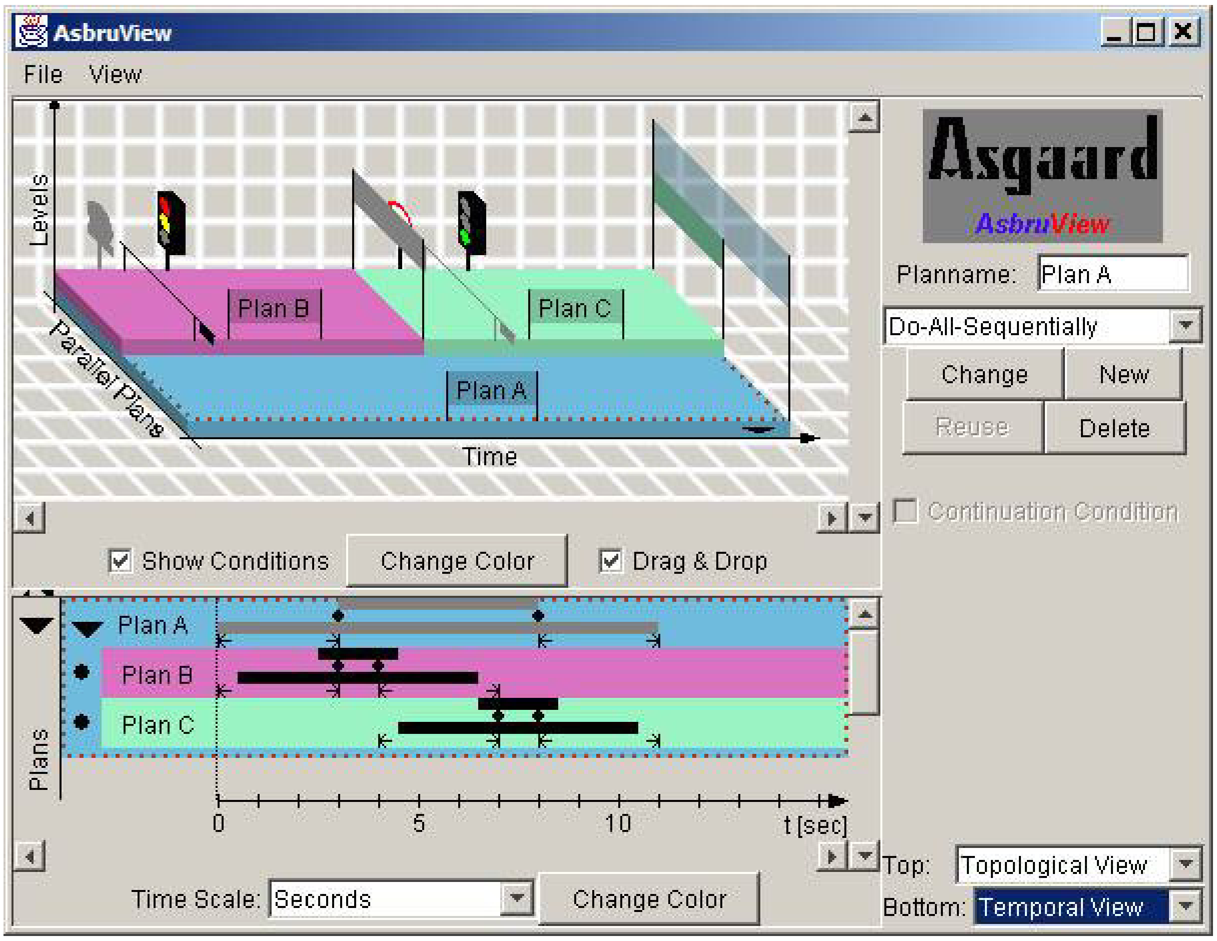


Figure 5.1: AsbruView - Metaphors of a Plan with Conditions

An *Asbru* plan is represented by a running track with a name sign ("Plan A" in Figure 5.1). The conditions of a plan are also visualized with the help of metaphors. The *filter* precondition is visualized by a "No Entry"-traffic sign, the *setup* precondition by a barrier, the *abort* condition by the red light on the traffic lights, the *suspend* condition by the yellow light, the reactivate condition by the green light and the *complete* condition by a finishing flag. These elements are drawn in grey when the corresponding condition is not defined and in colour otherwise.

To allow editing and visualization of all different aspects of Asbru, the user interface integrates three different views, which operate on the same Asbru guideline. Up to two of them can be shown at the same time, as seen in Figure 5.2:



Planname: Plan A

Time Scale: Seconds

Top: Topological View

Bottom: Temporal View

Figure 5.2: Screenshot of *Asbru*View

The topological view in the upper part of the window shows three plans illustrated with the metaphors described above. Plan B and plan C are subplans of plan A and will be executed sequentially. Plans can be easily re-ordered using drag-and-drop.

The temporal view in the lower part of the window uses the concept of *LifeLines*[1] to show the temporal dimension of the same plans. The two bars in each line depict their maximum and minimum duration using glyphs[2].

[1] *LifeLines* is a technique to visualize medical patient records and other personal histories by showing multiple timelines proposed by [Plaisant et al., 1996].

[2] Glyphs are "*graphical objects whose features reflect values, and therefore change their shape or size according to them*" [Kosara & Miksch, 2001].

The third available view is called SOPOView[1] and enables a definition of uncertainty in the time annotations (starting time, ending time and duration) of a plan. This additionally developed component ([Messner, 2000]) is similar to the temporal view, but uses two time axes instead that represent the begin and end times of an interval, respectively (Figure 5.3). Any point in this diagram represents an interval, specified by its start time (x-coordinate) and end time (y-coordinate).

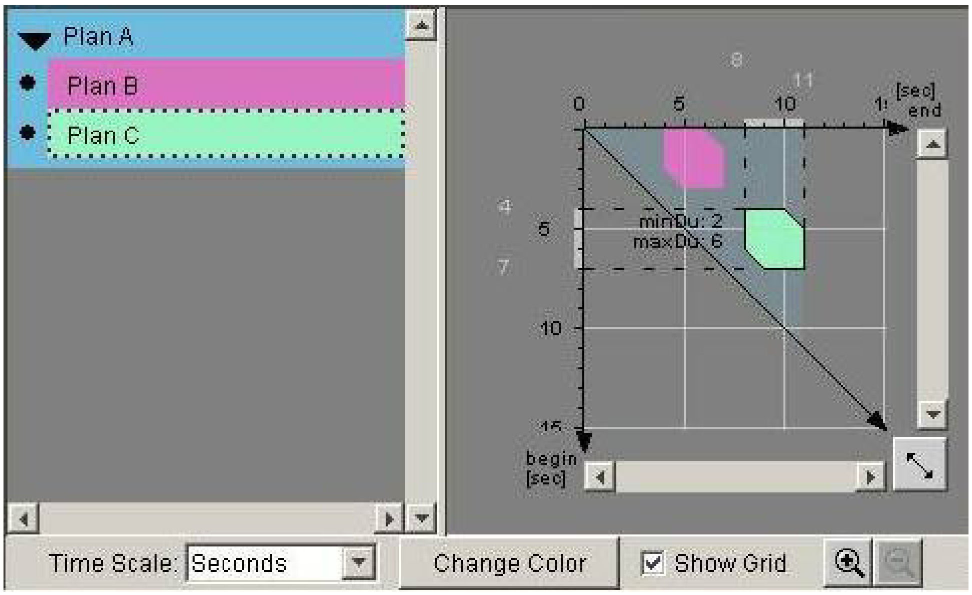


Figure 5.3: *Asbru*View - SOPOView

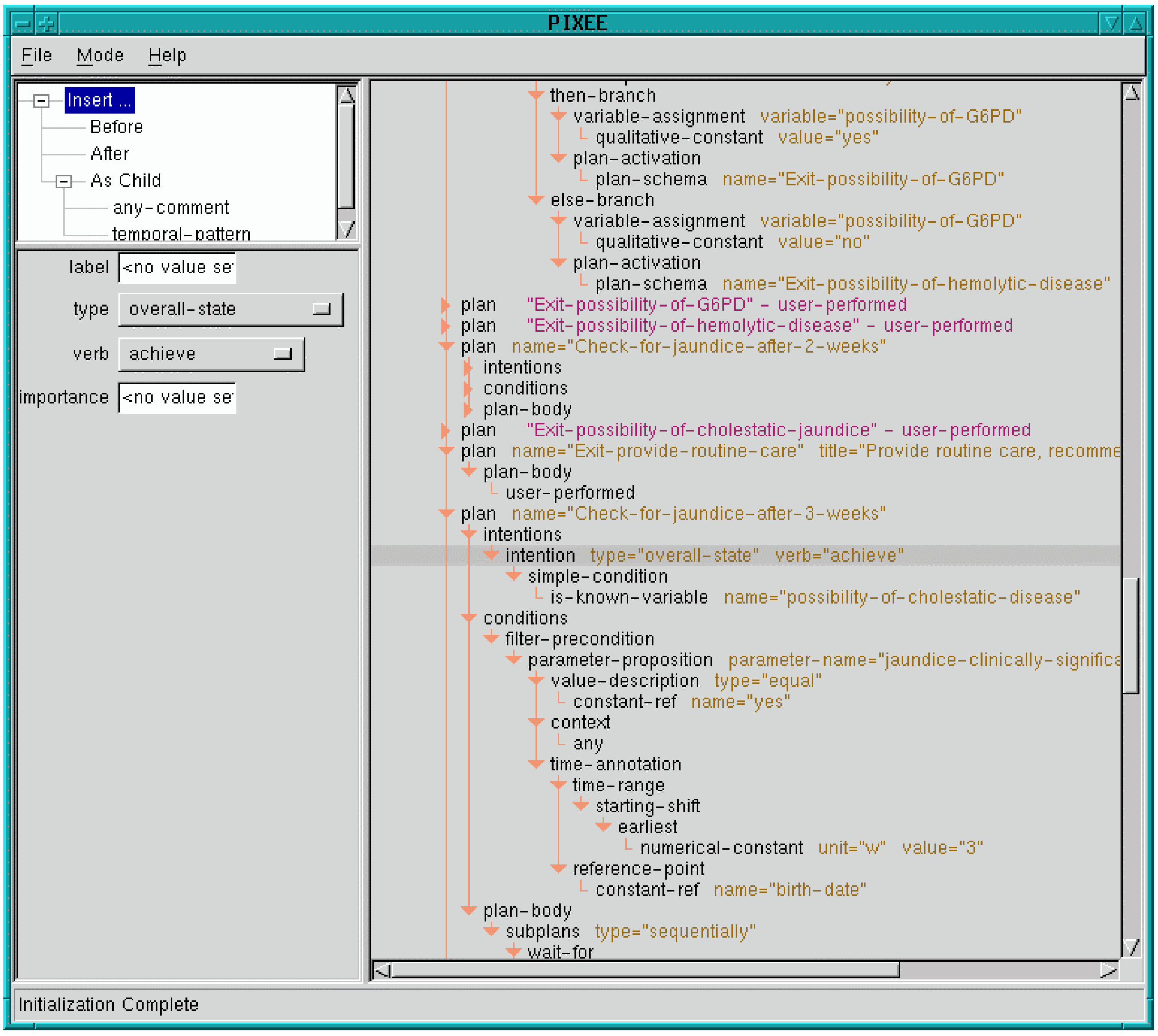
5.2 PIXEE

In opposition to *Asbru*View, **PIXEE** (*Pontifex* Intelligent XML Editor Extension) is a more general XML editor, which was also developed within the framework of the *Asgaard* project, to support low-level editing of Asbru guidelines. PIXEE (see [Kosara et al., 2002], p. 4-7) is built on top of *Pontifex* [2]([Kosara et al., 2000]), a utility, that takes the definition of an XML language and generates both a DTD and a Java class library.

PIXEE looks similar to other XML editors (see Figure 5.4), but primarily due to the usage of *Pontifex*, it offers some extra features. For example, it supports integer and float values, additional to the DTD attribute types. But the most advanced feature is the usage of data aggregation in the tree view: if a tree node is collapsed, information about its content, obtained through execution of pre-defined XPath[3] statements, is displayed instead of the element's attributes.

It is possible to use PIXEE completely independently from Asbru, but in the context of the Asgaard project, it is used by knowledge engineers to refine Asbru guidelines previously built with *Asbru*View or GMT.

[1] A SOPO (**S**et **o**f **P**ossible **O**ccurrences) specifies the temporal range where a given event can take place.
[2] *Pontifex* is Latin for "bridge builder"
[3] XML Path Language, or short XPath, is a language for addressing parts of an XML document, designed to be used by both XSLT and XPointer (see [URL #31]).



```
Insert ...
   Before
   After
   As Child
      any-comment
      temporal-pattern

     label  <no value se
      type  overall-state
      verb  achieve
importance  <no value se
```

```
            then-branch
              variable-assignment  variable="possibility-of-G6PD"
                qualitative-constant  value="yes"
              plan-activation
                plan-schema  name="Exit-possibility-of-G6PD"
            else-branch
              variable-assignment  variable="possibility-of-G6PD"
                qualitative-constant  value="no"
              plan-activation
                plan-schema  name="Exit-possibility-of-hemolytic-disease"
    plan   "Exit-possibility-of-G6PD" - user-performed
    plan   "Exit-possibility-of-hemolytic-disease" - user-performed
    plan  name="Check-for-jaundice-after-2-weeks"
      intentions
      conditions
      plan-body
    plan   "Exit-possibility-of-cholestatic-jaundice" - user-performed
    plan  name="Exit-provide-routine-care"  title="Provide routine care, recomme
      plan-body
        user-performed
    plan  name="Check-for-jaundice-after-3-weeks"
      intentions
        intention  type="overall-state"  verb="achieve"
          simple-condition
            is-known-variable  name="possibility-of-cholestatic-disease"
      conditions
        filter-precondition
          parameter-proposition  parameter-name="jaundice-clinically-significa
            value-description  type="equal"
              constant-ref  name="yes"
            context
              any
            time-annotation
              time-range
                starting-shift
                  earliest
                    numerical-constant  unit="w"  value="3"
              reference-point
                constant-ref  name="birth-date"
      plan-body
        subplans  type="sequentially"
          wait-for
```

Initialization Complete

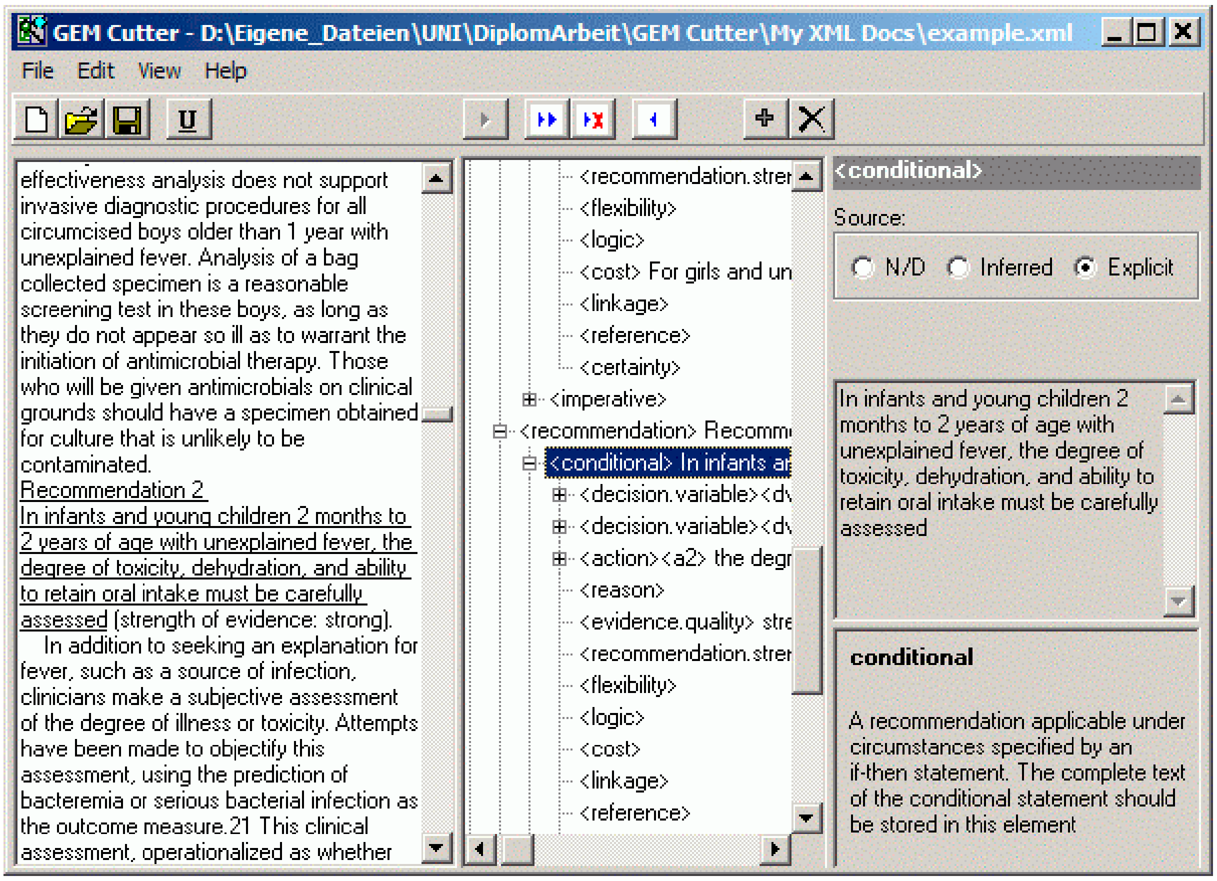
Figure 5.4: Screenshot of PIXEE

5.3 GEM Cutter

GEM Cutter is a tool that has been developed by the *Yale Center for Medical Informatics* to facilitate the transformation of guideline information into the GEM format (see chapter 4.2.2, p. 9). This is achieved by showing the original guideline document together with the corresponding XML document, as it has been done in the GMT.

The *GEM Cutter* window is subdivided into three parts (see Figure 5.5). The left part shows the original guideline text document from which the GEM file is being developed. Text can be easily copied from this view into elements in the GEM Tree view (on the right side). The selected text will become underlined afterwards.

The center part displays the contents of the GEM Document in a tree structure format. Each node of the tree represents a GEM Element. An element in this tree can be duplicated or deleted.


GEM Cutter - D:\Eigene_Dateien\UNI\DiplomArbeit\GEM Cutter\My XML Docs\example.xml
File
Edit
View
Help
U

effectiveness analysis does not support invasive diagnostic procedures for all circumcised boys older than 1 year with unexplained fever. Analysis of a bag collected specimen is a reasonable screening test in these boys, as long as they do not appear so ill as to warrant the initiation of antimicrobial therapy. Those who will be given antimicrobials on clinical grounds should have a specimen obtained for culture that is unlikely to be contaminated.
Recommendation 2
In infants and young children 2 months to 2 years of age with unexplained fever, the degree of toxicity, dehydration, and ability to retain oral intake must be carefully assessed (strength of evidence: strong).
In addition to seeking an explanation for fever, such as a source of infection, clinicians make a subjective assessment of the degree of illness or toxicity. Attempts have been made to objectify this assessment, using the prediction of bacteremia or serious bacterial infection as the outcome measure.21 This clinical assessment, operationalized as whether

```
<recommendation.stre
<flexibility>
<logic>
<cost> For girls and un
<linkage>
<reference>
<certainty>
⊞<imperative>
⊟<recommendation> Recomm
 ⊟<conditional> In infants ar
  ⊞<decision.variable><d
  ⊞<decision.variable><d
  ⊞<action><a2> the degr
   <reason>
   <evidence.quality> stre
   <recommendation.stre
   <flexibility>
   <logic>
   <cost>
   <linkage>
   <reference>
```

<conditional>

Source: ( ) N/D ( ) Inferred [✓ Explicit]

In infants and young children 2 months to 2 years of age with unexplained fever, the degree of toxicity, dehydration, and ability to retain oral intake must be carefully assessed

conditional

A recommendation applicable under circumstances specified by an if-then statement. The complete text of the conditional statement should be stored in this element

Figure 5.5: Screenshot of GEM Cutter

The right component shows additional information about the currently selected tree node and allows the editing of its text value.

For more detailed information about the usage of *GEM Cutter*, see the *GEM Cutter Manual* ([Polvani et al., 2000]).

Additionally, it should be said that the same ideas are behind *GEM Cutter* as behind the GMT. The main differences are briefly described hereafter:

- *GEM Cutter* does not store the connection between the selected text and the element that receives the text as its value. After re-opening the same text document again, all previously copied text will still be underlined, but this does not really help identifying which text passage belongs to which element. The GMT otherwise, maintains a bi-directional linkage between the original guideline document and the XML document.
- *GEM Cutter* works only with plain text and RTF documents. The GMT, on the other hand, accepts only HTML files.
- *GEM Cutter* was developed for the GEM language, so it is able to provide information about the current element. The GMT on the other side was designed to work with every XML language.
- Finally yet importantly, by choosing Java as programming language for the GMT, it runs on nearly every platform, unlike *GEM Cutter*, which is written in Visual Basic and runs therefore only on Windows.

5.4 PRO*forma* Tools

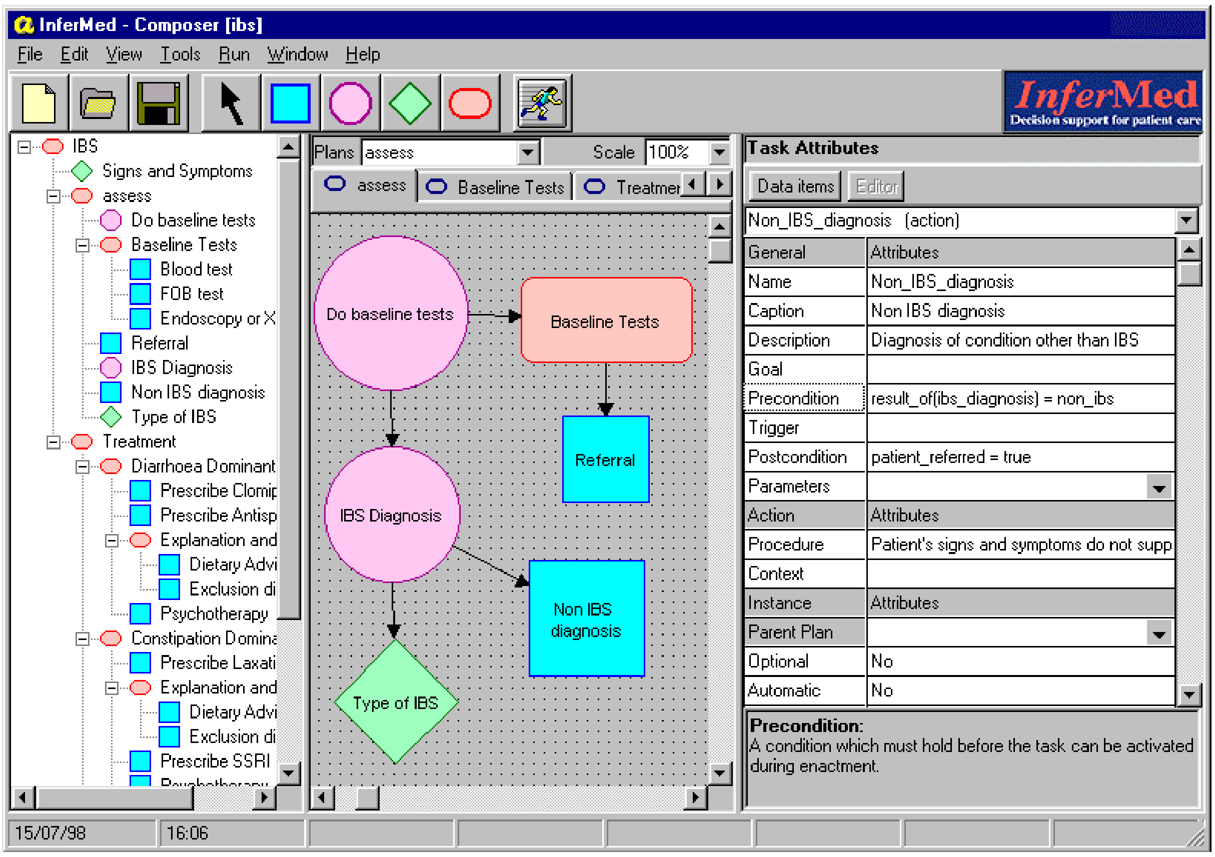
There are currently two technologies, which are based on the guideline representation language PRO*forma* (see chapter 4.2.7, p. 11): Since PRO*forma* is a relatively simple modelling method, they are both suitable for clinical professionals.

5.4.1 AREZZO

The first PRO*forma*-based system for authoring and executing clinical guidelines was **AREZZO** (see [URL #33]). It was designed and built by *Infer*Med and the *Advanced Computation Laboratory* (ACL) in London, and is being used by *Infer*Med to develop clinical guidelines in a form that can be interpreted by computers. The AREZZO system consists of two main modules: the *Composer* and the *Performer*.

The AREZZO *Composer* is a graphical editor (or knowledge-authoring tool), which uses PRO*forma* notation to capture the structure of a guideline and to generate an executable guideline representation. The building blocks used to construct a guideline of any level of complexity are the four PRO*forma* task types, each type being represented by its own icon. Data items and their properties are also defined using the composer.

Figure 5.6 shows the main screen of the graphical editor illustrating a part of a guideline for irritable bowel syndrome.



General	Attributes
Name	Non_IBS_diagnosis
Caption	Non IBS diagnosis
Description	Diagnosis of condition other than IBS
Goal	
Precondition	result_of(ibs_diagnosis) = non_ibs
Trigger	
Postcondition	patient_referred = true
Parameters	
Action	Attributes
Procedure	Patient's signs and symptoms do not supp
Context	
Instance	Attributes
Parent Plan	
Optional	No
Automatic	No

Precondition:
A condition which must hold before the task can be activated during enactment.

Figure 5.6: Screenshot of the AREZZO Composer

On the left hand side is a tree view of the protocol being edited. In the centre is a graphical representation of the component tasks of one specific task, and on the right hand side is the Task Attributes grid, where the attribute values of each task may be viewed and edited. This grid may also be switched for the user to view and edit the protocol's data definitions. The toolbar along the top provides buttons for opening and saving protocols and for creating additional tasks.

The second tool, the AREZZO Performer can be used to validate, execute and debug PRO*forma* guidelines.

5.4.2 TALLIS

TALLIS ([Steele & Fox, 2002]) is a new suite of software tools to support authoring and publishing clinical knowledge applications over the WWW. According to the information available ([URL #34]), TALLIS is the only toolset currently available for publishing web-enabled guidelines. Since TALLIS is also developed by the *Advanced Computation Laboratory*, the included editor (Figure 5.7) looks very similar to the AREZZO *Composer*.

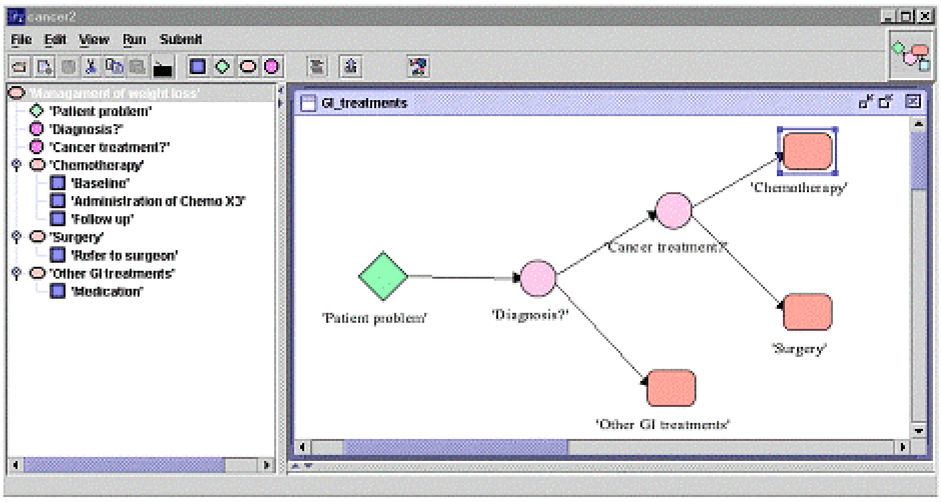


Figure 5.7: Screenshot of the TALLIS Editor

After creating a PRO*forma* guideline with the TALLIS editor, it can be published as a special web application, called *publet*, on a web server. A publet consists of HTML forms and optional Java applets and allows the execution of the guideline through any web browser.

5.5 Guide-X

Guide-X ([Svatek et al., 2000]) was developed at the *Laboratory for Intelligent Systems*, University of Economics, Praha. Although Guide-X is just a methodology without any implementation yet, it fits into this chapter, because its approach is similar to that of the Guideline Markup Tool.

Most approaches to guideline computerization assume that the formalization is based on extensive interaction between the domain expert and the knowledge engineer. Such a one-step formalization process, remains quite troublesome, and requires a mixture of expertise: medical expertise, general knowledge engineering skills, as well as expertise in handling the particular target representation.

Breaking the formalization process down into multiple steps can be helpful both in separating the different types of expertise required into relatively independent parts, and in making the process more transparent and allowing for easier verification and correction of misinterpretations.

The Guide-X (Guideline formalization based on XML) methodology consists of five steps (see Figure 5.8). For the output of the three intermediate steps, new XML languages have been developed by the Guide-X project.

The first step produces a XHTML file, either by converting an existing HTML guideline or by creating a new one from scratch. The second step expects this XHTML file as input, removes unnecessary markup and adds simple semantic information. The output is a GLML-S (Guideline Markup Language - Simple) XML file. The third step takes the GLML-S file, refines it by adding new elements and produces a GLML-R (Guideline Markup Language - Rich) file. The fourth step does not further refine the markup of the original document, but completely restructure the GLML-R file in favour of systematic ordering of the comprised knowledge and produces a GLKL (Guideline Knowledge Language) file. The last step results in a non-XML file based on OCML (Operational Concept Modelling Language). The final output can then be used for further processing.

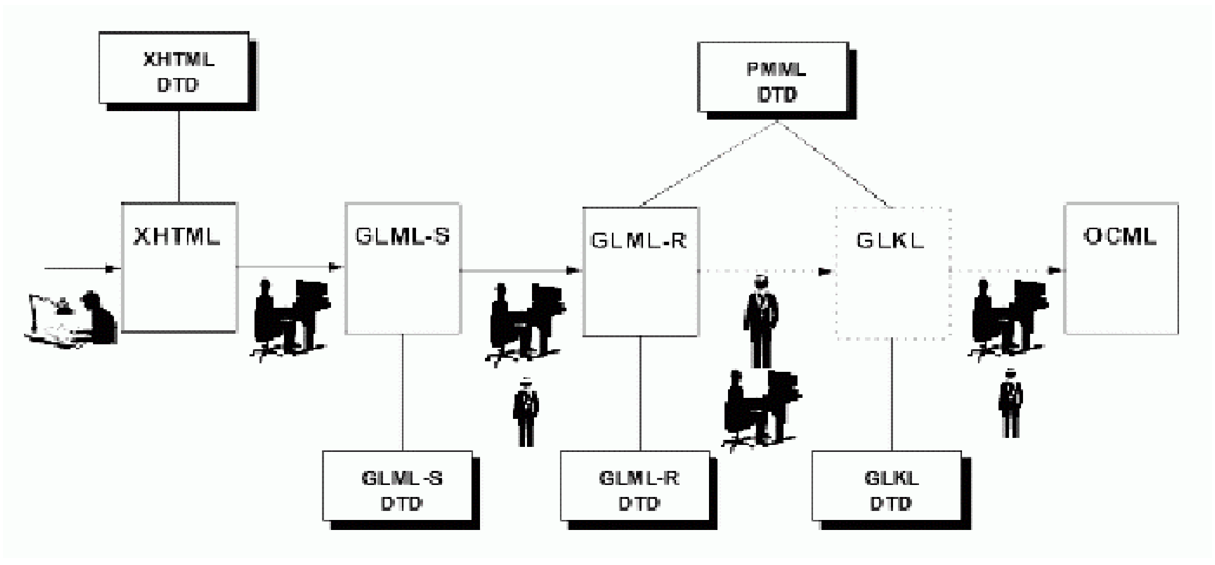


Figure 5.8: Steps of the Guide-X Methodology

6 Summary of presented projects

Several different projects, which focus on guideline modelling and acquisition, have been presented so far. This chapter tries to give an overview of the different approaches. It is divided into two parts, since it is impossible to compare a guideline representation model with a guideline tool. A comparison study of guideline models

6.1 Guideline Modelling Methods

Although differences in the main purpose exist, such as the GLIF approach desires to function as an interchange format and PRO*forma*'s intention is to facilitate decision making, all want to create an electronic representation of a guideline which can be further used for validation and computer-assisted execution of the guideline.

Several studies and papers (for example, [Roomans, 2001]) have been done on the comparison of different guideline modelling methods. Comparing all the different aspects and describing the advantages and disadvantages of the eight methods presented so far would go beyond the scope of this thesis. For a detailed comparison of the different guideline modelling methods, see the comparison study ([Peleg et al., 2002a] and [Peleg et al., 2002b]) initiated by *OpenClinical* (see [URL #36]).

6.2 Guideline Tools

*Asbru*View (chapter 5.1, p. 12) is certainly the best tool for visualizing the different aspects of *Asbru* guidelines. Additionally, it is also a very simple editor, because of its heavy use of graphical metaphors. But it does not support all *Asbru* elements and it provides no means to translate a guideline into its *Asbru* representation.

PIXEE (chapter 5.2, p. 14) has been called "*A Slightly More Useful XML Editor*", which says quite a lot about its purpose. The most useful utilization of PIXEE is to use it for refining or extending an existing guideline.

As elaborated in chapter 5.3 (p. 15), GEM Cutter is similar to the *Guideline Markup Tool*. Like the GMT, it provides two views of the same guideline, but unlike the GMT it does not allow links between the original guideline and the GEM representation.

The graphical editors of the two Pro*forma* systems look very similar. They are both using the same symbols for the four types of tasks (see Figure 4.1) in their graphical editors, thus providing a straightforward overview of a Pro*forma* guideline. The difference is that the TALLIS system supports publishing and executing a Pro*forma* guideline on World Wide Web, whereas the AREZZO system is designed for local use only.

II Design and Implementation

7 The Guideline Markup Tool

The *Guideline Markup Tool* (see Figure 7.1) helps a knowledge engineer to transform the content of an HTML file, representing a clinical guideline, into an *Asbru* XML file. Furthermore, it allows the definition of links between the original guideline and the Asbru representation, which gives the user the possibility to find out from where a certain value in the XML file comes from.

This chapter first describes the ideas behind the GMT. Then the user interface and the main features are explained. Finally, some implementation's details are presented.

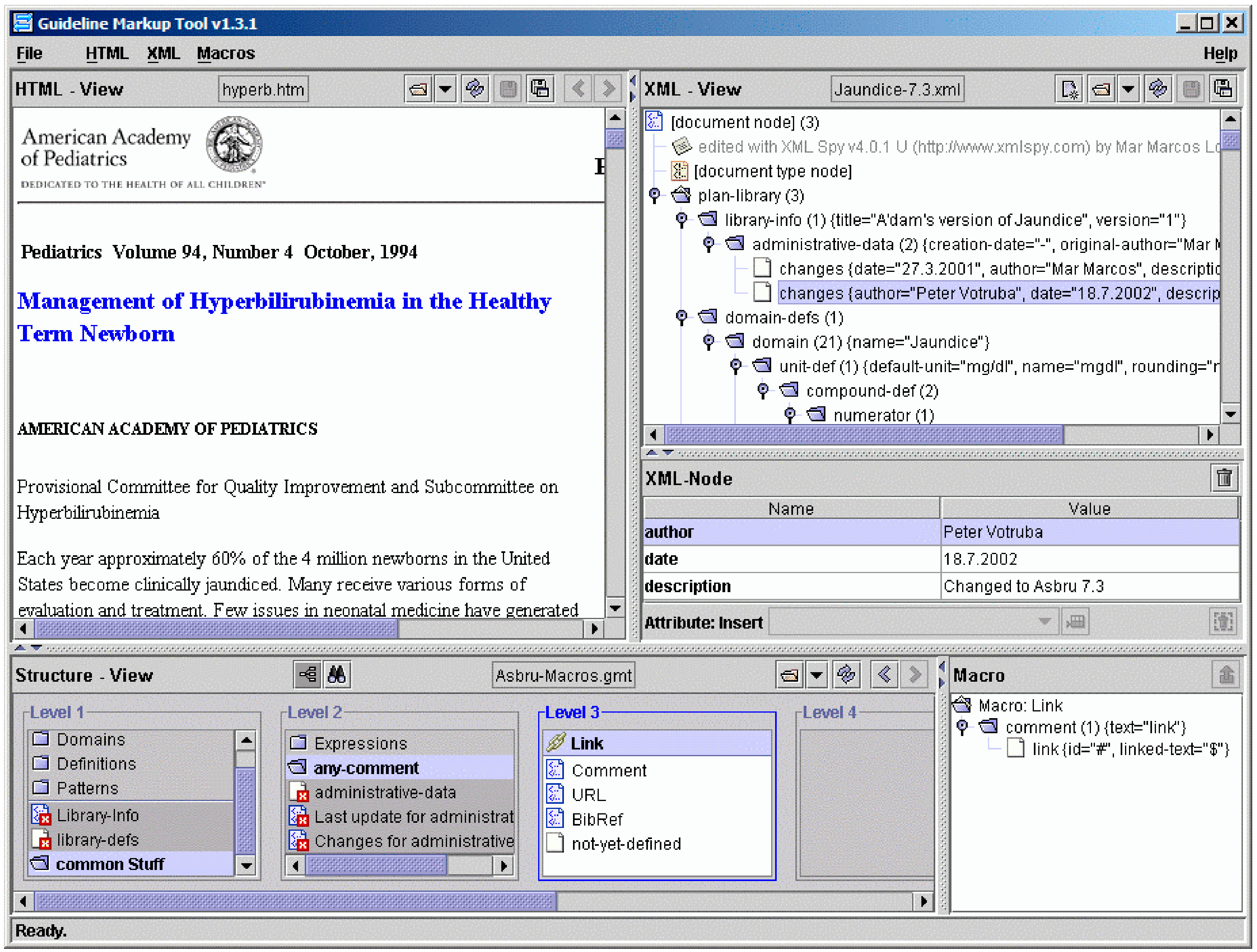
7.1 Motivation

A special-purpose guideline tool has already been developed within the *Asgaard* project: *Asbru*View (see chapter 5.1, p. 12). Through the usage of metaphors, it is suitable for the daily work of professional clinicians. However, physicians would not normally create an *Asbru* guideline from scratch. This is the job of knowledge engineers, which should have experience in both, the medical domain and *Asbru*.

A new tool was needed that supports knowledge engineers in translating a guideline into the *Asbru* language. Since many clinical guidelines are published over the World Wide Web using the HTML format, this tool should provide an HTML view. Showing this view together with the content of the *Asbru* XML file, similar to *GEM-Cutter* (chapter 5.3, p. 15), would facilitate the translation task. This leads us to the main idea of the *Guideline Markup Tool*.

The GMT should enable the knowledge engineer to create links between the original guideline and its XML representation. Later, if someone wants to know the origin of a specific value in the XML file, the GMT can be used to jump to the correlating point in the HTML file where the value is defined.

Additional to the linking feature, the second main point of the GMT is the usage of macros. A macro combines several XML elements, which are usually used together. Thus, using macros facilitates creating and extending *Asbru* XML files. Information about macros can be found in chapter 8.



American Academy of Pediatrics
DEDICATED TO THE HEALTH OF ALL CHILDREN

Pediatrics Volume 94, Number 4 October, 1994

Management of Hyperbilirubinemia in the Healthy Term Newborn

AMERICAN ACADEMY OF PEDIATRICS

Provisional Committee for Quality Improvement and Subcommittee on Hyperbilirubinemia

Each year approximately 60% of the 4 million newborns in the United States become clinically jaundiced. Many receive various forms of evaluation and treatment. Few issues in neonatal medicine have generated

```
[document node] (3)
  edited with XML Spy v4.0.1 U (http://www.xmlspy.com) by Mar Marcos L
  [document type node]
  plan-library (3)
    library-info (1) {title="A'dam's version of Jaundice", version="1"}
      administrative-data (2) {creation-date="-", original-author="Mar M
        changes {date="27.3.2001", author="Mar Marcos", descriptio
        changes {author="Peter Votruba", date="18.7.2002", descrip
    domain-defs (1)
      domain (21) {name="Jaundice"}
        unit-def (1) {default-unit="mg/dl", name="mgdl", rounding="r
          compound-def (2)
            numerator (1)
```

Name	Value
author	Peter Votruba
date	18.7.2002
description	Changed to Asbru 7.3

Level 1	Level 2	Level 3	Level 4
Domains	Expressions	Link	
Definitions	any-comment	Comment	
Patterns	administrative-data	URL	
Library-Info	Last update for administrat	BibRef	
library-defs	Changes for administrative	not-yet-defined	
common Stuff			

```
Macro: Link
  comment (1) {text="link"}
    link {id="#", linked-text="$"}
```

Figure 7.1: GMT Window

7.2 User Interface

According to the requirements presented above, the user interface is designed to show the contents of the HTML file, the XML file and the macros file together in one window. Therefore, the GMT window is divided into three main parts – Figure 7.2 shows a screenshot of the main window after the first start.

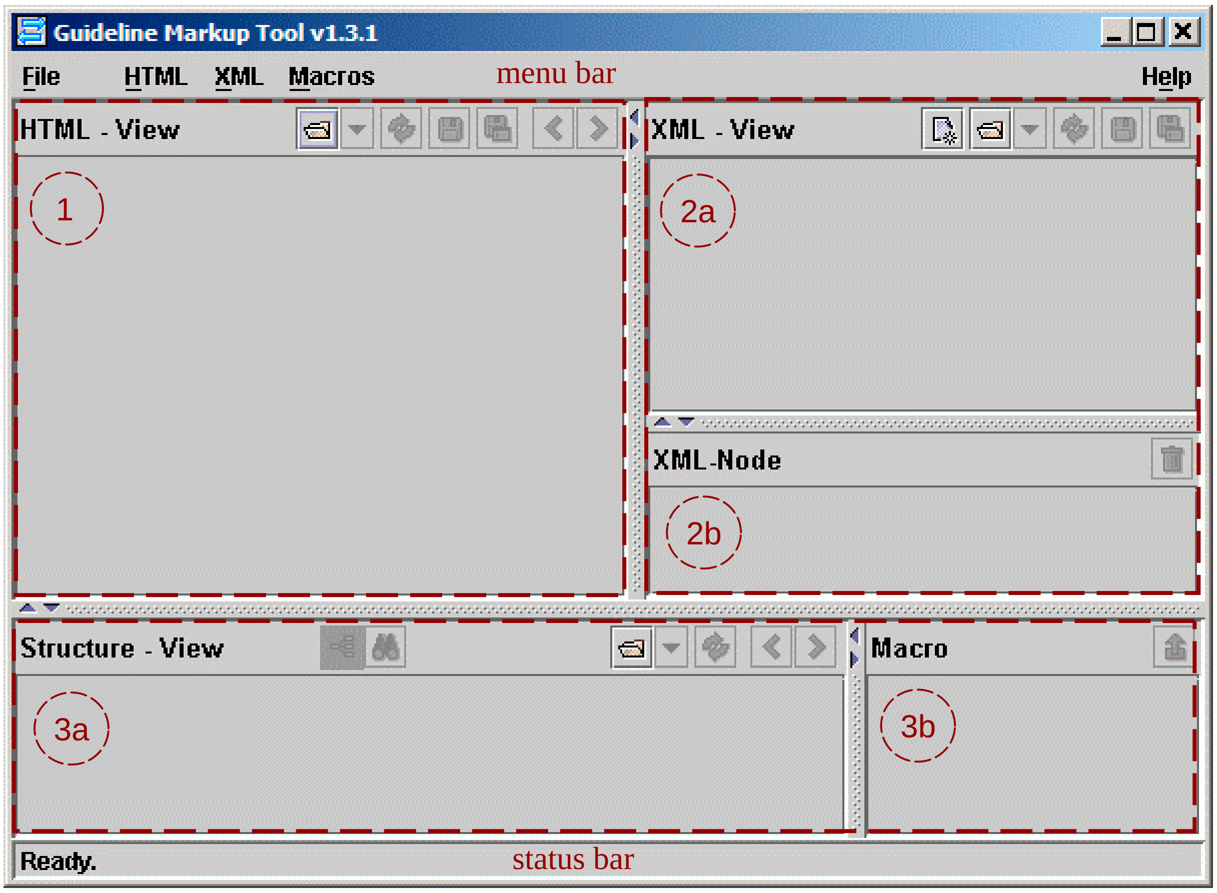


Figure 7.2: Parts of GMT Window

The upper left component (1) shows the contents of the HTML file. The XML part consists of a hierarchical view of the XML file (2a) and a detail view of the current XML node (2b). The macros part contains a view of the macros structure (3a) and a pre-view of the currently selected macro (3b).

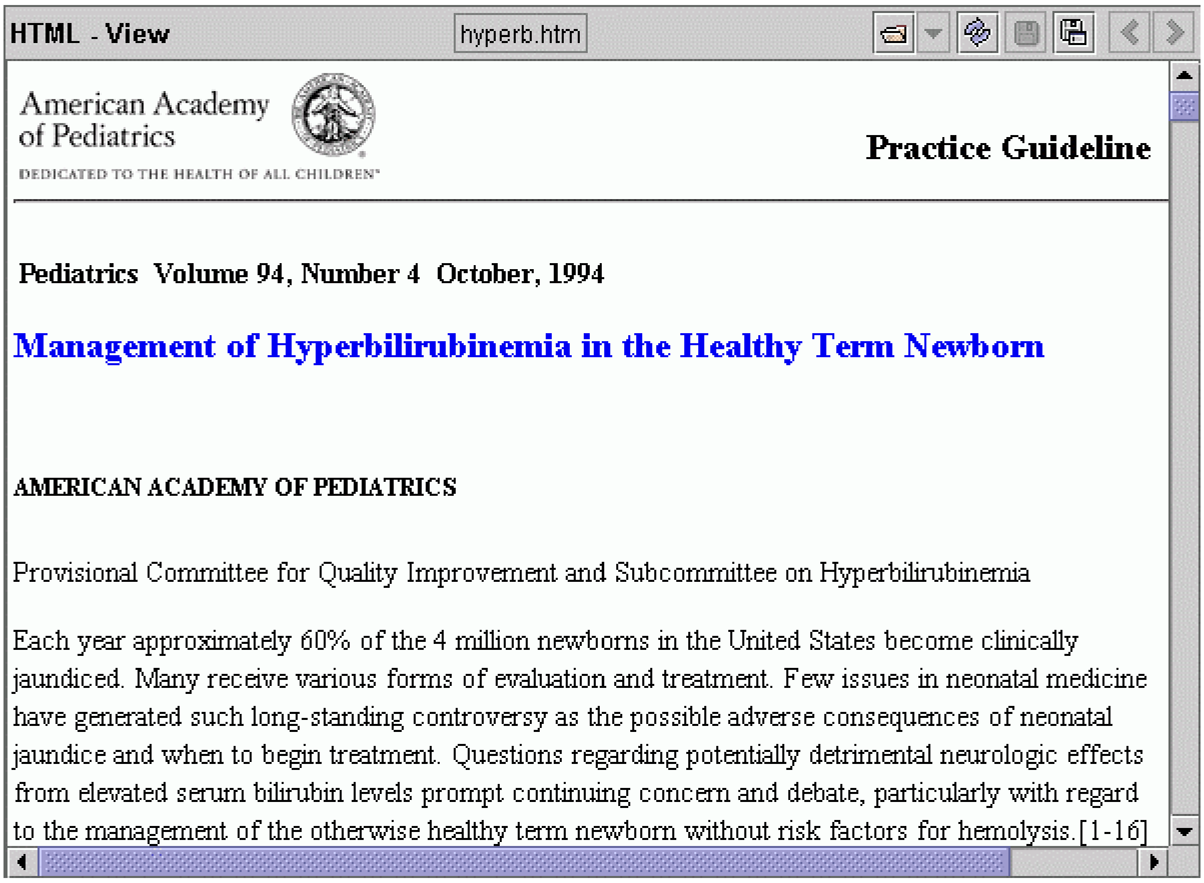
The proportions of the components can be changed easily by dragging one of the horizontal (▒) or vertical (▒) dividers. If a component is not needed for a specific task, it can be fully minimized by clicking on the corresponding little arrow (▲/▼ or ◀/▶) on one of the adjoining dividers.

At the top of the GMT window is the menu bar, which allows access to all features of the GMT and at the bottom the status bar, which shows information about the current action.

The following sub-chapters take a closer look at these three main parts.

7.2.1 HTML View

The HTML View shows the content of an HTML document (see Figure 7.3 for a sample screenshot). Basically, the HTML View is like a simple web browser - it supports internal and even external links.



American Academy of Pediatrics
DEDICATED TO THE HEALTH OF ALL CHILDREN

Practice Guideline

Pediatrics Volume 94, Number 4 October, 1994

Management of Hyperbilirubinemia in the Healthy Term Newborn

AMERICAN ACADEMY OF PEDIATRICS

Provisional Committee for Quality Improvement and Subcommittee on Hyperbilirubinemia

Each year approximately 60% of the 4 million newborns in the United States become clinically jaundiced. Many receive various forms of evaluation and treatment. Few issues in neonatal medicine have generated such long-standing controversy as the possible adverse consequences of neonatal jaundice and when to begin treatment. Questions regarding potentially detrimental neurologic effects from elevated serum bilirubin levels prompt continuing concern and debate, particularly with regard to the management of the otherwise healthy term newborn without risk factors for hemolysis.[1-16]

Figure 7.3: GMT – HTML View

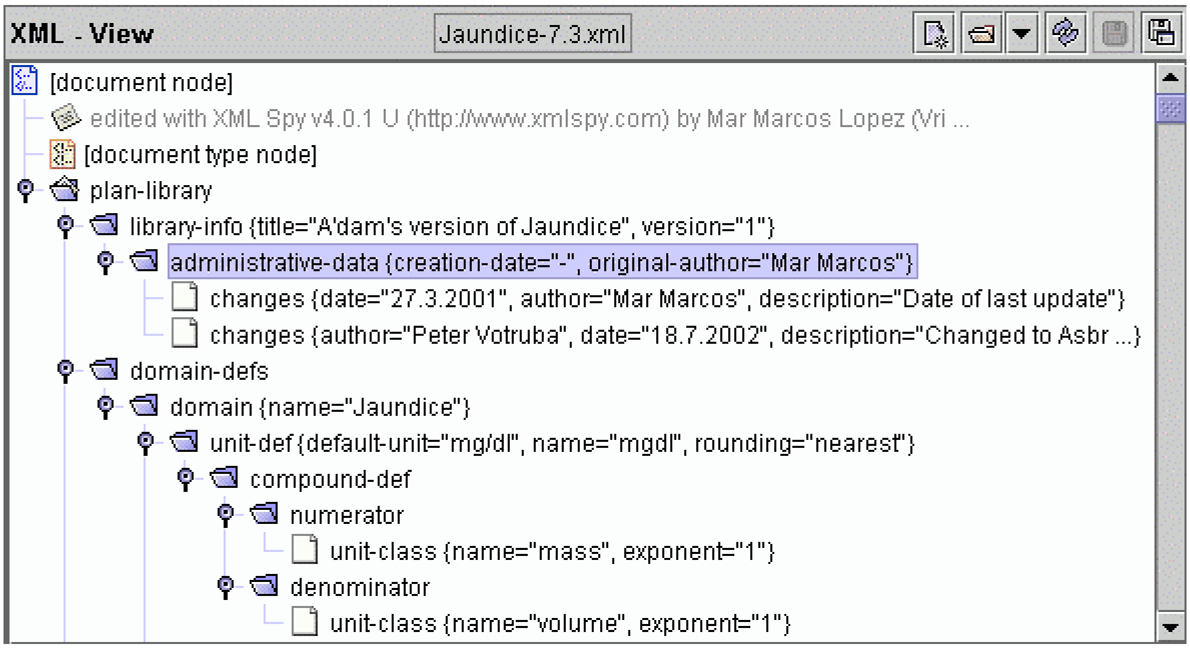
Additionally, it enables – together with the XML-View – the linkage between text passages and XML elements. Such links are represented similar to standard HTML links in the HTML-View, except that the colour is cyan instead of blue.

In comparison with current web browsers, the HTML-View supports only fundamental web standards: HTML 3.2 and most of the basic features of Cascading StyleSheets (CSS), but neither JavaScript nor DHTML[1]. Nonetheless, this is sufficient for our needs, because this view is intended to show only HTML guidelines, which usually contain no complex formatting.

[1] Dynamic HTML refers to technologies that allow documents to be changed after their initial display, without server access, through user interaction and client-side scripting.

7.2.2 XML View

The hierarchical content of the XML file is shown in a tree view (see Figure 7.4). The other component of the XML part is an extra view that shows detailed information about the currently selected node.



XML - View   Jaundice-7.3.xml

```
[document node]
  edited with XML Spy v4.0.1 U (http://www.xmlspy.com) by Mar Marcos Lopez (Vri ...
  [document type node]
  plan-library
    library-info {title="A'dam's version of Jaundice", version="1"}
      administrative-data {creation-date="-", original-author="Mar Marcos"}
        changes {date="27.3.2001", author="Mar Marcos", description="Date of last update"}
        changes {author="Peter Votruba", date="18.7.2002", description="Changed to Asbr ...}
    domain-defs
      domain {name="Jaundice"}
        unit-def {default-unit="mg/dl", name="mgdl", rounding="nearest"}
          compound-def
            numerator
              unit-class {name="mass", exponent="1"}
            denominator
              unit-class {name="volume", exponent="1"}
```

Figure 7.4: GMT – XML View

Every node in the tree view corresponds to a node in the XML document. The different types of XML nodes, which may appear in the XML-View, are explained subsequently.

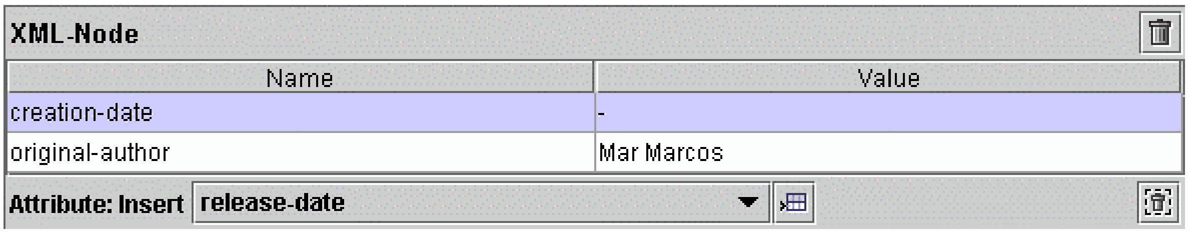
The document node () is the root node of the tree view and contains the *document type node*, the *document element* and optional comment nodes. The document type node () represents the DOCTYPE-statement of the XML document, which specifies the DTD file.

If a branch element is collapsed, it is shown as otherwise as . A leaf element is visualized with or with if it is a link element. The top element is called *document element* and is represented with , if it is expanded. Beside the element name, a list of attributes of this element is shown in curly brackets. The attributes can be changed using the detail view below the XML view (see Figure 7.5, below).

XML comments (defined with "`<!-- ... -->`" in the XML document) are represented with . If the text of the comment is too long for the XML-View, it can be seen in the detail view below. The remaining types of XML nodes – text (), CDATA () and processing instruction nodes () – are not used in *Asbru*, but still supported by the XML-View.

The XML-View supports the deletion of the currently selected node using the button or the "Remove Node" - menu item.

Figure 7.5 shows the detail view for the currently selected element node in Figure 7.4.



XML-Node

Name	Value
creation-date	-
original-author	Mar Marcos

Attribute: Insert release-date

Figure 7.5: GMT - Detail View of an Element Node

This table allows simple editing of the attribute values. It shows required attributes with a bold font and default values with an italic font. The combo box below the table shows the missing attributes of the current element, which can be inserted using the button. Optional attributes can be removed using the button or the "Remove Attribute" - menu item.

7.2.3 Macros Structure View

Although the macros in a macros file form a hierarchical data structure, they are not represented using a tree view, because it would not fit into the GMT window (see Figure 7.2) due to its shape. Instead, a new horizontal component for the visualization of hierarchical data has been developed. It uses list boxes to show the contents of each group of macros (left part of Figure 7.6).

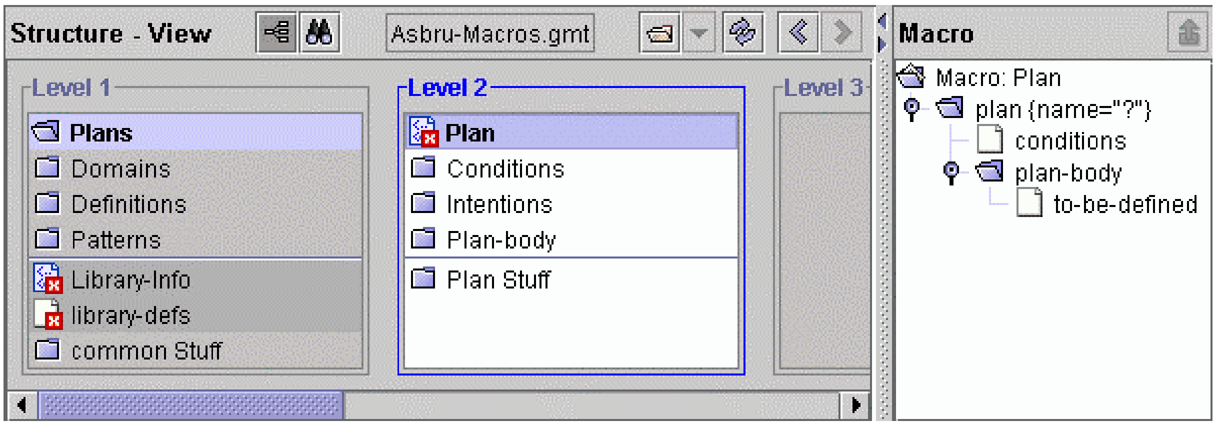


Figure 7.6: GMT - Macros Structure View with Macro Pre-view

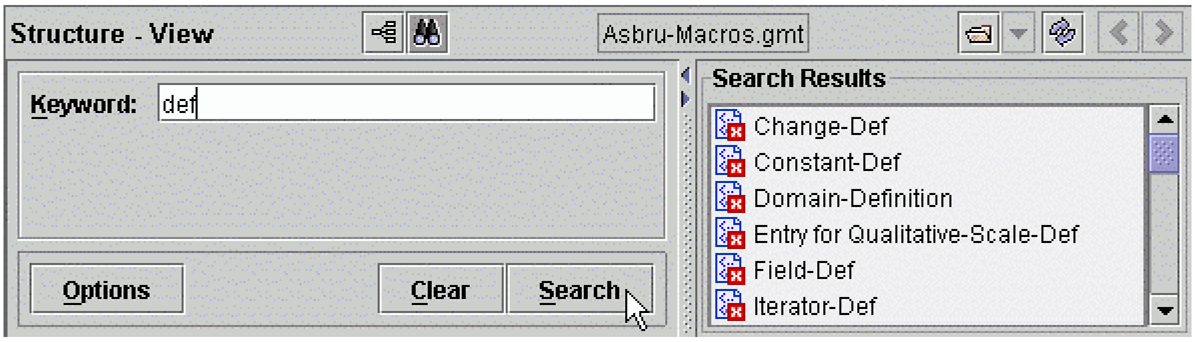
If a list item, representing a group, is selected, the content of that group is shown in the next list box. If a list item, representing a macro, is selected (like in the figure above), a preview of that macro is shown next to the Structure-View.

As far as possible, I have tried to use the same icons in the structure view as in the XML-View: Groups are represented with the folder icons (,) used in the XML view. The icons for macros (), single elements () and macros that contain a link () are also taken from the XML view.

Each macro and element, which are not allowed to be inserted in the currently selected node of XML-View, are marked with .

Only the representation of group links is new – for example: Temporal Patterns.

To facilitate the locating of a specific macro, a search function has been included (Figure 7.7).


Structure - View

Asbru-Macros.gmt

Keyword: def

Options

Clear

Search

Search Results

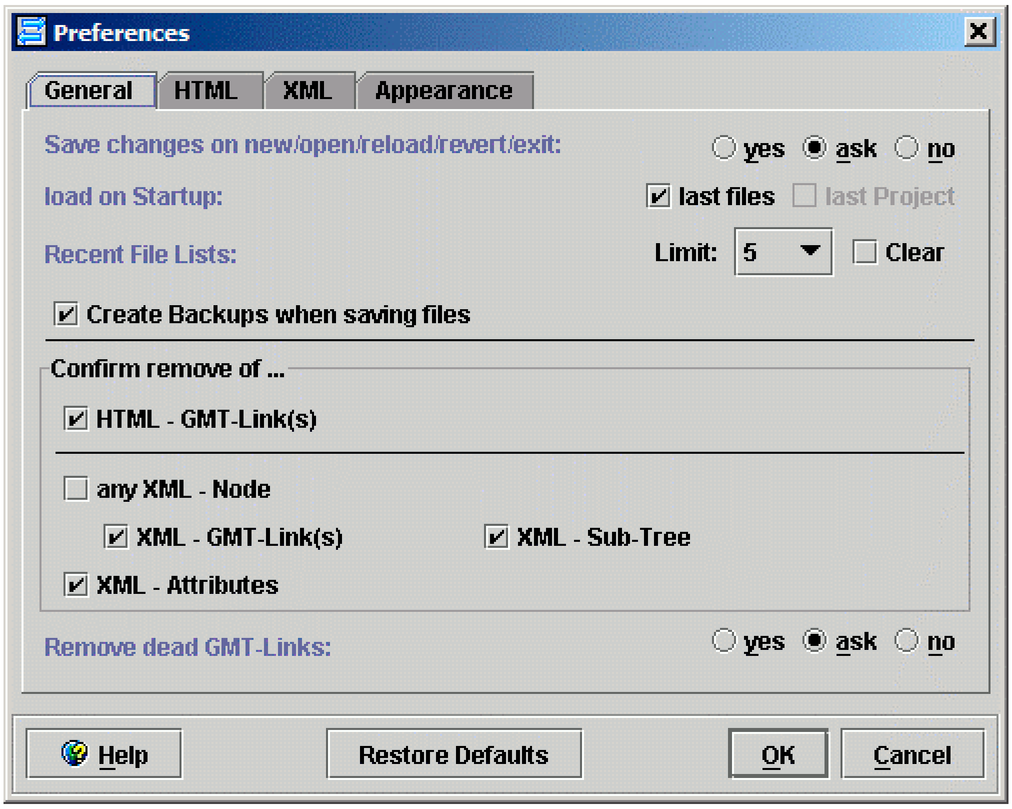
- Change-Def
- Constant-Def
- Domain-Definition
- Entry for Qualitative-Scale-Def
- Field-Def
- Iterator-Def

Figure 7.7: GMT - Macros Search View

After clicking on the "Search"-button, all macros, which have the entered keyword in their names, will be displayed in the right list box.

7.2.4 Preferences Dialog

The GMT preferences dialog is shown when the menu item "Preferences..." in the "File" menu has been selected. It offers many useful options, separated into four categories. The first tab (see Figure 7.8) provides general options, for example a group of options, which specify if the removal of XML elements should be confirmed.


Preferences

General | HTML | XML | Appearance

Save changes on new/open/reload/revert/exit: yes [✓ ask] no

load on Startup: [✓ last files] last Project

Recent File Lists: Limit: 5 ☐ Clear

[✓ Create Backups when saving files]

Confirm remove of ...

[✓ HTML - GMT-Link(s)]

☐ any XML - Node

[✓ XML - GMT-Link(s)] [✓ XML - Sub-Tree]

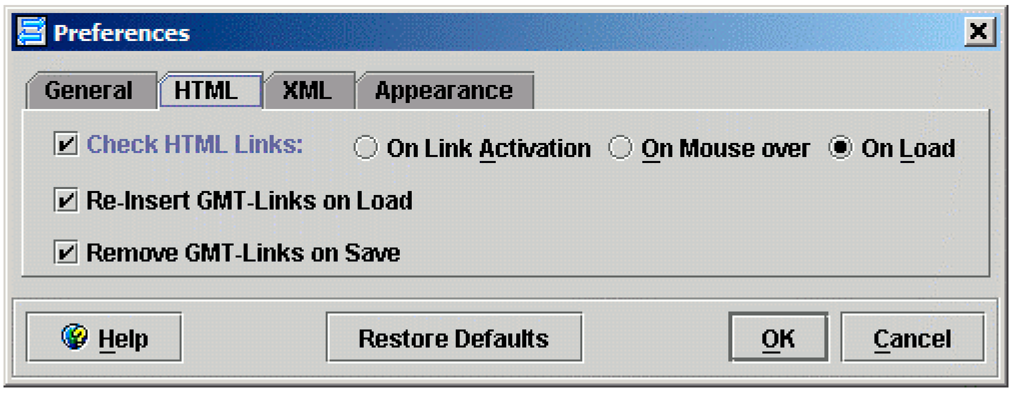
[✓ XML - Attributes]

Remove dead GMT-Links: yes [✓ ask] no

Help | Restore Defaults | OK | Cancel

Figure 7.8: GMT - Preferences Dialog (General)

The second tab (see Figure 7.9) contains options regarding the HTML view. The user can specify if the special GMT links should be visible in an external browser.


Preferences

General | HTML | XML | Appearance

[✓] Check HTML Links: ○ On Link Activation ○ On Mouse over [✓ ● On Load]

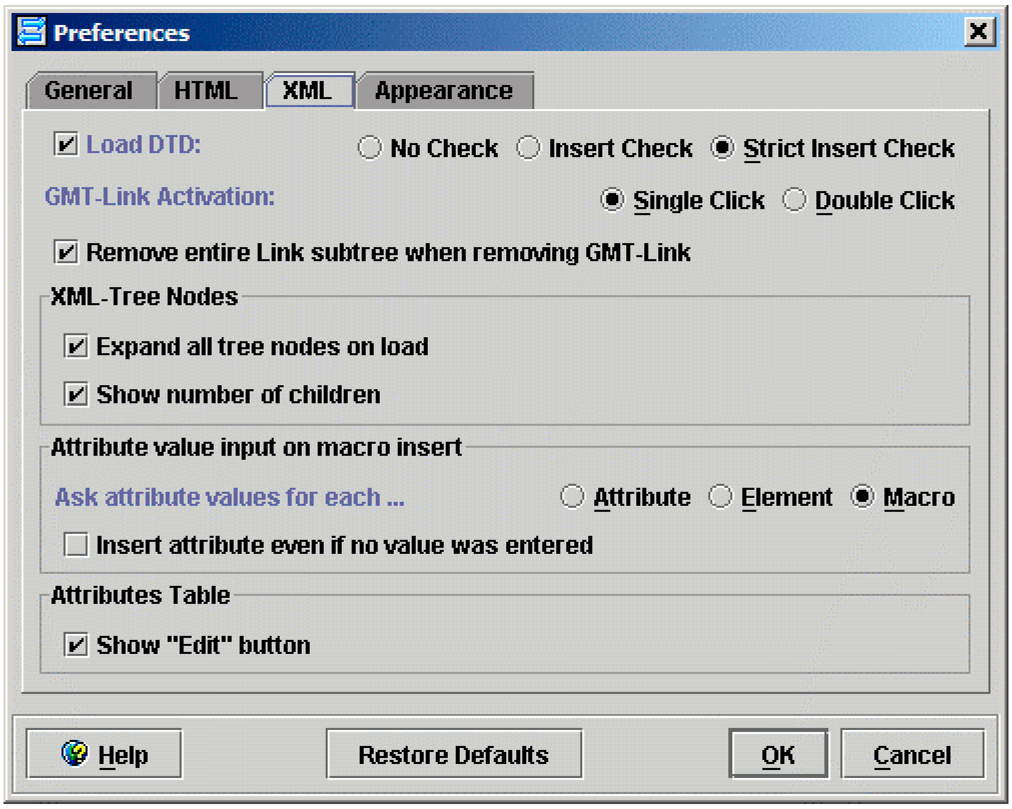
[✓] Re-Insert GMT-Links on Load

[✓] Remove GMT-Links on Save

Help | Restore Defaults | OK | Cancel

Figure 7.9: GMT- Preferences Dialog (HTML)

The third tab (see Figure 7.10) offers various XML-related options. Amongst other things, the user can specify if GMT links in the XML view could be activated by a single click or by double click.


Preferences

General | HTML | XML | Appearance

[✓] Load DTD: ○ No Check ○ Insert Check [✓ ● Strict Insert Check]

GMT-Link Activation: [✓ ● Single Click] ○ Double Click

[✓] Remove entire Link subtree when removing GMT-Link

XML-Tree Nodes

[✓] Expand all tree nodes on load

[✓] Show number of children

Attribute value input on macro insert

Ask attribute values for each ... ○ Attribute ○ Element [✓ ● Macro]

[ ] Insert attribute even if no value was entered

Attributes Table

[✓] Show "Edit" button

Help | Restore Defaults | OK | Cancel

Figure 7.10: GMT - Preferences Dialog (XML)

7.3 Main Features

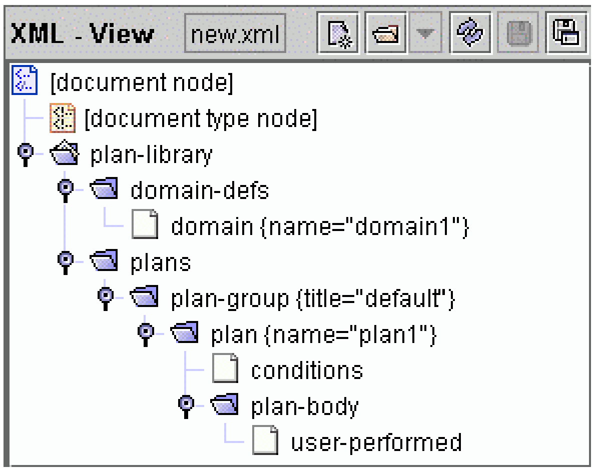
All features are available through the main menu bar. Most of the items in the HTML, XML and Macros menus have corresponding buttons on the toolbars (for example "Open…" or "Save"). Some of the others appear also in popup-menus, which are shown on a right-click in the HTML view, XML view or macro pre-view (for example "Remove Node" in the XML popup-menu). All menu items (in the menus of the main menu bar and in the popup-menus) and all toolbar buttons have tool tips with short descriptions. Additionally, a longer description is displayed in the status bar while the mouse is being moved over a menu item or toolbar button.

This chapter presents the main features of the GMT. A detailed and complete explanation of all available features can be found in the "GMT User Manual" ([Votruba, 2003]).

7.3.1 Starting Translation of a Guideline

To start the translation of a guideline, the HTML file containing the guideline have to be loaded into the HTML view first, either by clicking on the button in the toolbar or by selecting the "Open…" menu item in the "HTML" menu.

The second step is to create a new *Asbru* XML file, either by clicking on the button or by selecting the "New…" menu item in the "XML" menu. After choosing a new file name, the XML view generates the skeletal structure of a valid *Asbru* file (see Figure 7.11) using an internal template.



```
[document node]
  [document type node]
  plan-library
    domain-defs
      domain {name="domain1"}
    plans
      plan-group {title="default"}
        plan {name="plan1"}
          conditions
          plan-body
            user-performed
```

Figure 7.11: GMT – XML View with new *Asbru* XML File

In future versions of the GMT, there will be the possibility to choose from different templates, when creating a new XML file.

Finally, the Asbru macros file have to be loaded in the macros structure view, either by clicking on the button in the toolbar or by selecting the "Open…" menu item in the "Macros" menu.

7.3.2 Inserting Macros

First, the XML element that should become the target of the macro insertion, have to be selected in the XML view. In Figure 7.12, this is the element *domain*. Afterwards, the macros structure view indicates which macros are insertable.

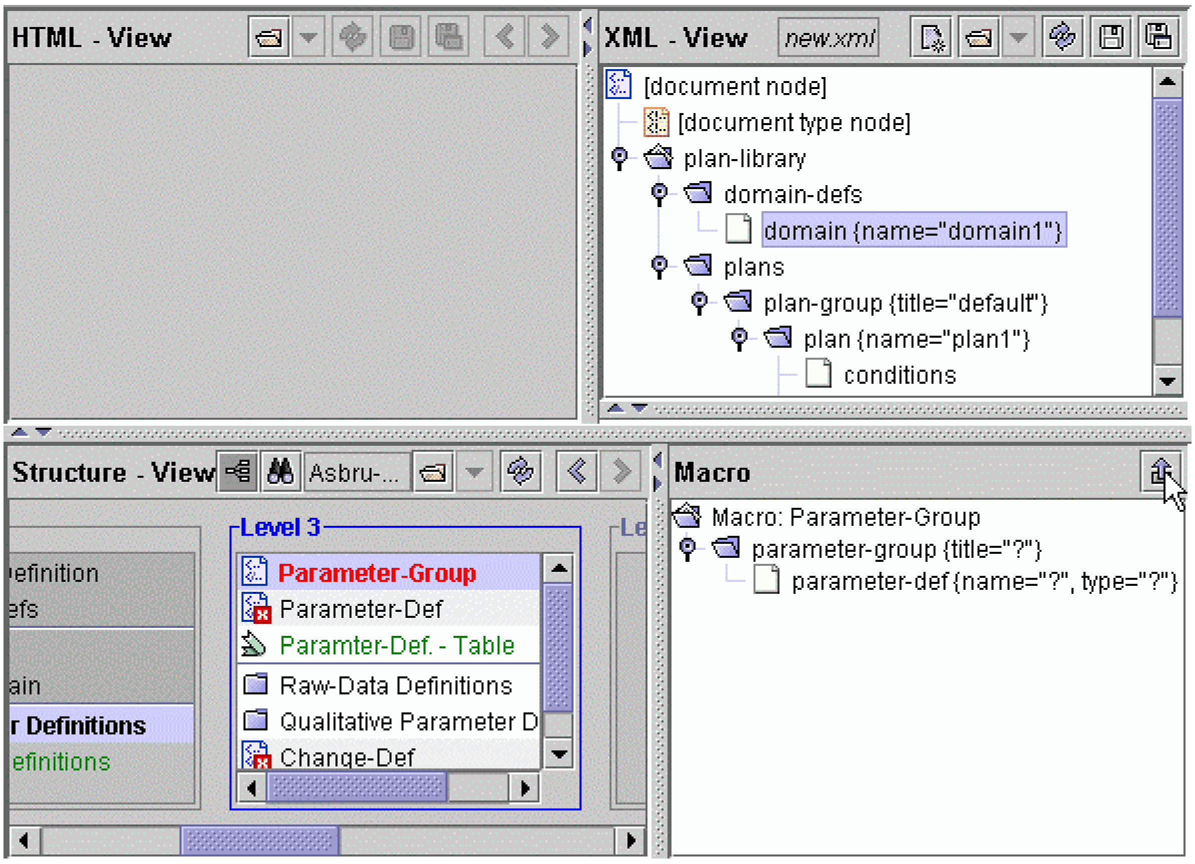


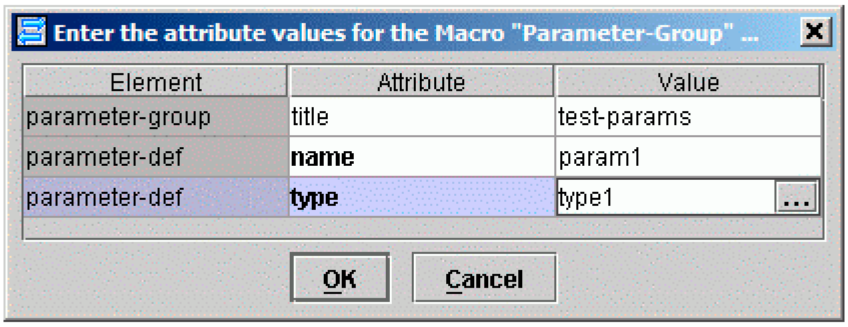
Figure 7.12: GMT Window - before Macro Insertion

In our case, only the macro *Parameter-Group* is eligible – the other two visible macros are not allowed at this time, as indicated by the ☒ on their icons.

After the macro *Parameter-Group* is selected, its colour turns to red. This is a hint that indicates that this macro is not complete, that means that one or more child elements are missing, according to the DTD, and should be added later. In our case the element *parameter-def* needs one extra child element, which should be inserted afterwards.

Additionally, the content of the macro is shown in the macro pre-view next to the structure view.

The next step is to choose "Insert Macro" either from the menu or by clicking on the ⇪ button. In our case, the macro is defined to ask for the attribute values of its elements and so, a input dialog (see Figure 7.13) will be shown, where the attribute values can be entered.

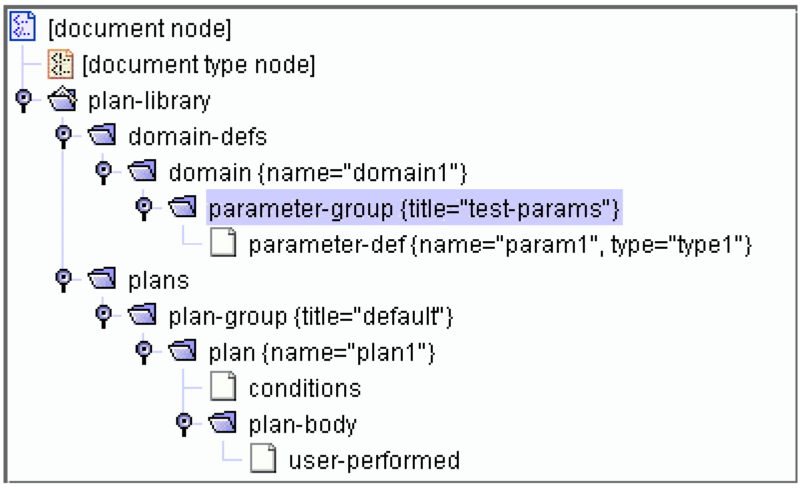


Enter the attribute values for the Macro "Parameter-Group" ...

Element	Attribute	Value
parameter-group	title	test-params
parameter-def	name	param1
parameter-def	type	type1

Figure 7.13: GMT - Attribute Input during Macro Insertion

After closing the dialog, the macro will be inserted. The resulting XML tree can be seen in Figure 7.14.



```
[document node]
  [document type node]
  plan-library
    domain-defs
      domain {name="domain1"}
        parameter-group {title="test-params"}
          parameter-def {name="param1", type="type1"}
    plans
      plan-group {title="default"}
        plan {name="plan1"}
          conditions
          plan-body
            user-performed
```

Figure 7.14: GMT - Result of Inserting Macro

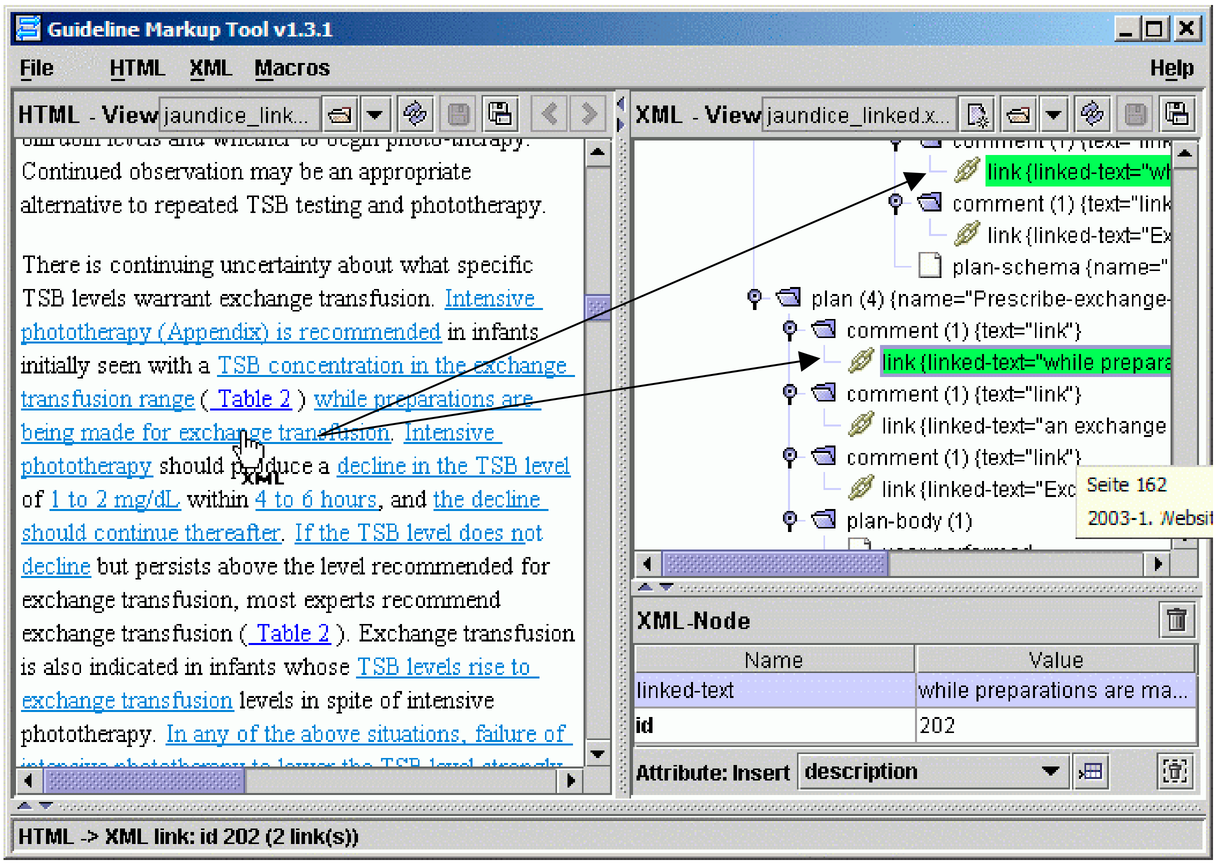
7.3.3 Link Features

Inserting a link is similar to inserting a macro. First, the XML element, which should be linked to the HTML document, has to be selected in the XML-View. Then a macro in the Macros Structure-View, which contains a link definition (is indicated by icon) have to be chosen. Then the text passage in the HTML-View, which should become the other end of the connection, have to be selected. After inserting the link macro, by clicking on the button, the selected text will be underlined, like an HTML link.

Each link is identified by a unique number, which is stored as an attribute of the link element. It is possible to have multiple link endpoints with the same number on each side, which means that they all belong together.

If the cursor is being moved over an HTML→XML link, it changes to XML or, if it has no corresponding link on the other side, to XML.

If the user clicks on a link in the HTML- or XML-View, all link endpoints with the same id-number will be highlighted on the other side (Figure 7.15).


Guideline Markup Tool v1.3.1
File HTML XML Macros Help
HTML - View jaundice_link...
XML - View jaundice_linked.x...
XML

Continued observation may be an appropriate alternative to repeated TSB testing and phototherapy.

There is continuing uncertainty about what specific TSB levels warrant exchange transfusion. Intensive phototherapy (Appendix) is recommended in infants initially seen with a TSB concentration in the exchange transfusion range (Table 2) while preparations are being made for exchange transfusion. Intensive phototherapy should produce a decline in the TSB level of 1 to 2 mg/dL within 4 to 6 hours, and the decline should continue thereafter. If the TSB level does not decline but persists above the level recommended for exchange transfusion, most experts recommend exchange transfusion (Table 2). Exchange transfusion is also indicated in infants whose TSB levels rise to exchange transfusion levels in spite of intensive phototherapy. In any of the above situations, failure of

```
link {linked-text="wh
comment (1) {text="link
  link {linked-text="Ex
plan-schema {name="
plan (4) {name="Prescribe-exchange-
  comment (1) {text="link"}
    link {linked-text="while prepara
  comment (1) {text="link"}
    link {linked-text="an exchange
  comment (1) {text="link"}
    link {linked-text="Exc
  plan-body (1)
```

Seite 162
2003-1. Websit

XML-Node

Name	Value
linked-text	while preparations are ma...
id	202

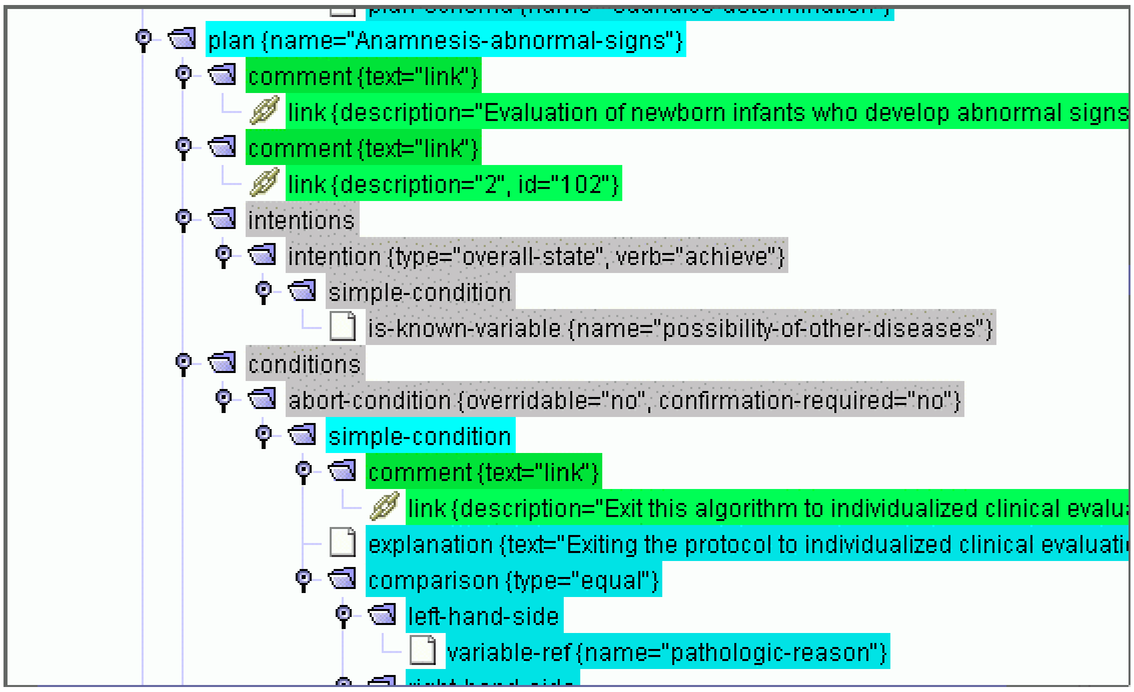
Attribute: Insert description

HTML -> XML link: id 202 (2 link(s))

Figure 7.15: GMT - activation of GMT link

In both views, links can be deleted. If a link is deleted and there are no more other links with the same id-number in the same view, the corresponding links on the other side become "dead links". The user will be asked, if such dead links should also be deleted, depending on the "Remove dead links" option in the preferences dialog (see Figure 7.8).

To visualize the distribution of links in an *Asbru* XML document, the "Link Visualization" feature (see Figure 7.16) can be used. The link elements will be highlighted with green and each element of a subtree, which contains a link, will be coloured in cyan, whereas all other nodes in the XML view get a gray background.



```
plan {name="Anamnesis-abnormal-signs"}
  comment {text="link"}
    link {description="Evaluation of newborn infants who develop abnormal signs
  comment {text="link"}
    link {description="2", id="102"}
  intentions
    intention {type="overall-state", verb="achieve"}
      simple-condition
        is-known-variable {name="possibility-of-other-diseases"}
  conditions
    abort-condition {overridable="no", confirmation-required="no"}
      simple-condition
        comment {text="link"}
          link {description="Exit this algorithm to individualized clinical evalu
        explanation {text="Exiting the protocol to individualized clinical evaluati
        comparison {type="equal"}
          left-hand-side
            variable-ref {name="pathologic-reason"}
```

Figure 7.16: GMT – Link Visualization

7.4 Implementation

Choosing Java as the programming language for the GMT entails some valuable advantages.

First, the same version of the GMT runs on every system for which a Java virtual machine is available (and this means on nearly every system). Additionally, developing with Java is very comfortable for the programmer, because of the great range of excellent Java IDEs (integrated development environments) and powerful programming libraries, which can be used.

One of the mentioned libraries is Swing, integrated in the Java runtime environment, which offers many useful classes for building graphical user interfaces. For example, the tree view, used to show the contents of an XML document, and the HTML view are adapted *Swing* components.

On the other hand, many parts of the application make extensive use of XML technologies. The XML document is internally represented using classes based on the popular DOM (Document Object Model) standard.

Additionally, the powerful XSLT (eXtensible Stylesheet Language Transformations) technology has been utilized by using Java's TRaX (Transformation API for XML) standard. To simplify the process of transforming the contents of a macro to the target XML language (*Asbru*), a XSLT stylesheet has been written and integrated. Moreover, the user is able to view a coloured and therefore better readable version of the XML source code through using another built-in XSLT stylesheet.

The GMT is designed to work with any DOM compliant XML parser (for example *Xerces*) and any TraX compliant XSLT stylesheet processor (for example *Xalan*).

Additionally, it uses IBM's XML4j, the only free XML library (I have found) that provides DTD-specific features. Thus, the GMT is able to tell the user which macro is allowed to be inserted into the current element and what attributes can be added to an element according to the DTD.

8 The Macros File

Macros files used in the GMT are written in XML. A special XML language, the GMT macros language, has been developed for this purpose. It allows the simple definition and structuring of macros. A macro is a structure of XML elements that are often used together. It can be seen as a template, which can be easily reused during authoring of guidelines. The smallest reasonable macro consists of one element with one attribute. The size of a macro is not limited – it may even contain more than one top element, but it is not recommended to define a macro with more than ten elements, because such a big structure would decrease the insight and usability of this macro.

This chapter first explains the structure of a macros file and describes the most important elements. Then the *Asbru* macros file, which was developed for this thesis, will be presented.

8.1 Structure of a Macros File

The root element of a macros file is `<gmt>`. Each macros file is divided into three main parts (see Figure 8.1), which are briefly described subsequently.

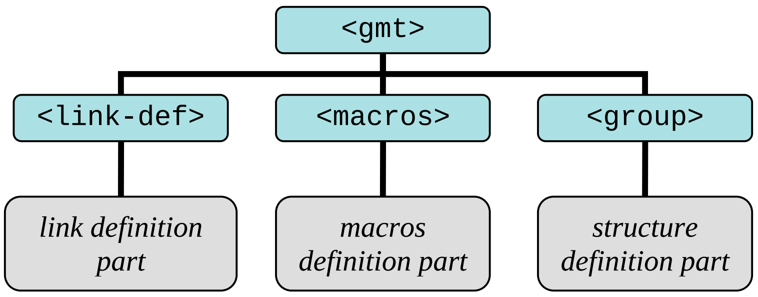


Figure 8.1: Basic structure of a macros file

For information about all possible elements of a macros file, see appendix C (p. 52), which contains the source code of the Document Type Definition file (`gmt.dtd`).

8.1.1 Link definition part

In the current version of the GMT, links are inserted using a special macro, usually named "Link". Additionally, other macros may also contain one or more links. Therefore, the link is defined centralized. Each macro, which should include a link, could refer to the central link definition using the `<link-location/>` element. Thus, the "Link"-macro is always defined in the following way:

```
<macro name="Link" description="inserts a GMT link">
    <link-location/>
<macro/>
```

For an example of the link definition part, see chapter 8.2.3.

8.1.2 Macros definition part

The macros definition part consists of a list of all macros, each enclosed by a `<macro>` element. A macro has a unique name and an optional description. The content of a macro is defined using the following elements:

- `<element-ref>` refers to an existing element of the target XML language.
- `<attribute-ref>` defines an attribute and its value for the element of the enclosing `<element-ref>`.
- A macro may also contain other macros using the `<macro-ref>` element.
- As shown above, the `<link-location/>` element can be used to refer to the central link definition.

When a macro is used in the GMT (see chapter 7.3.2), the elements of the macro are transformed to the target XML language (in our case, Asbru).

`<element-ref>` and `<attribute-ref>` produce an element or an attribute, respectively, with the specified name.

Example:

Macros Source Code		Target XML Code

`<element-ref name="domain">`
`    <attribute-ref name="name"`
`                   value="Domain1"/>`
`</element-ref>`
	leads to	`<domain name="Domain1"/>`

There are four special characters, which may be used in the value of a `<attribute-ref>'s 'value'` attribute instead of a pre-defined attribute value:

- "**?**": If a "?" is used as an attribute value, the user will be asked for that value.

  Example:

  ```
  <macro name="Domain">
      <element-ref name="domain">
          <attribute-ref name="name" value="?"/>
      </element-ref>
  </macro>
  ```

  If the macro `"Domain"` will be inserted, the user will be asked for the value of the attribute `"name"`. Thus, this macro leads to the following target XML code:

  `<domain name="`*entered value*`"/>`.

- If text is selected in the HTML-View, every "$" character in an attribute's value will be replaced by the selected text.
- A "%" character in an attribute's value will be replaced by the current date.
- The "#" character will be replaced by a sequential number and is intended to be used only in links.

During the insertion of a macro all included `<macro-ref>` and `<link-location/>` elements are replaced by the transformed contents of the specified macros or the link definition part (`<link-def>`).

8.1.3 Structure definition part

The structure definition part defines the hierarchical structure, which is presented in the Macros View in the GMT.

Following elements can be used in the structure definition part:

- A `<group>` element is used to combine other elements – the content of a group will be shown in a list box in the Macros-View.
- A `<macro-ref>` element refers to an existing macro.
- Single elements without children and attributes can be modelled without macros using `<single-element-ref>'s`.
- A `<separator/>` element produces a dividing line in the corresponding list box in the Macros-View.
- A `<label/>` element can be used to show a descriptive label in the list box, for example below a dividing `<separator/>`.

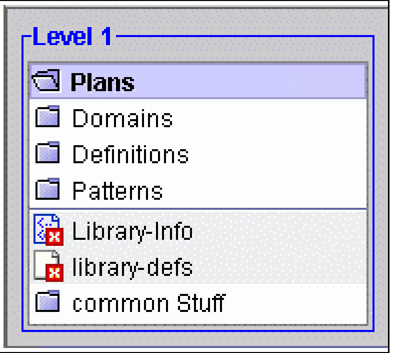
Example**:**

Macros File

```
<group name="root">
  <group name="Plans"/>
  <group name="Domains"/>
  <group name="Definitions"/>
  <group name="Patterns"/>
  <separator/>
  <macro-ref name="Library-Info"/>
  <single-element-ref name="library-defs"/>
  <group name="common Stuff"/>
</group>
```

will be shown as

Macros Structure View



8.2 The *Asbru* Macros File

A macros file for *Asbru* (see chapter 4.1, p. 8) has been developed within the framework of this thesis. During the development of the macros file, I have oriented myself towards the Asbru 7.3 version of the "*Management of Hyperbilirubinemia in the Healthy Term Newborn*" guideline, originally created by the *Protocure* project (see [URL #17]).

It could not be said that the development of this macros file is finalized – it is neither optimal nor complete. It was a first attempt und should be seen as the basis for future work in this domain - it is certainly a good starting point for further improvements or developments.

This chapter explains the design goals of the Asbru macros file, gives an overview of its structure and presents some sample macros.

8.2.1 Design Goals

There are some initial considerations that were relevant for the development of the Asbru macros file:

- ➢ The Structure definition part of the macros file should definitely not mirror the structure of the Asbru DTD. Instead, the structuring of the macros should be orientated towards the usage of the macros.

- Macros should be grouped by their relevance (so all macros, which were often used, are at higher levels) and by their field of application (for example, all macros, which deal with Asbru plans, should be in the same sub-tree). This would make the locating of macros easier when using the GMT.
- Groups should not be overloaded - especially because the content of the groups is shown in the GMT in relatively small list boxes. A maximum number of ten items per group is recommended.
- Macros should not be defined multiple times. Instead, all related macros should be defined in a central group. Every group, which needs such a macro, gets a link to this group (`<group-link/>`).
- There is no need to model all Asbru elements – this could be done in future versions of the Asbru macros file. Primarily, Asbru elements, which are necessary for beginning a new Asbru file, should be modelled.

8.2.2 Hierarchical Structure

The macros structure, which is provided by the *Asbru* macros file, could be divided into three main parts: *Plans*, *Domains* and *Definitions*. Figure 8.2 shows a graphical overview of the macros structure containing nearly all groups.

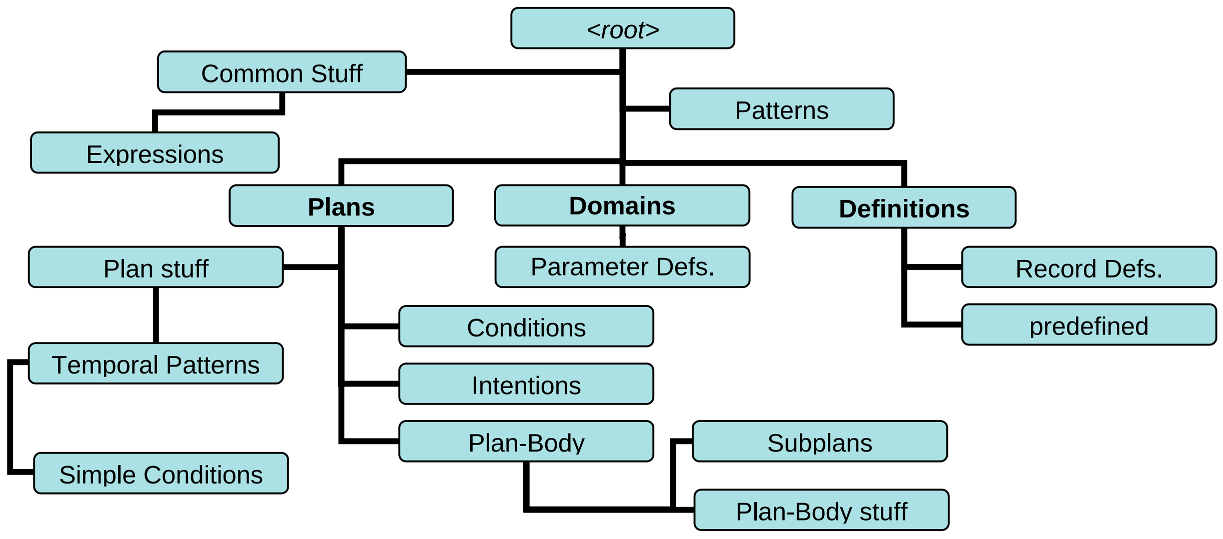


Figure 8.2: Structuring of the Asbru macros

The *Plans* subtree contains macros that are necessary to create *Asbru* plans. It has extra subtrees for the most important aspects of plans: conditions, intentions and plan-body. The other plan-specific macros are combined in the group *Plan stuff*.

The *Domains* group contains macros, which are used to define domains as well as macros for domain-specific definitions, like parameter definitions.

The *Definitions* part combines general definition macros, which can be inserted in domains, plans and in the rarely used `library-defs` element.

Besides those three main parts, there are two other top-level groups: *Patterns* and *Common Stuff*. The former refers to patterns, which are common to clinical guidelines. In the current version, there are only table-pattern macros in this group. The latter covers common macros, like expressions, which can be inserted into various *Asbru* elements.

Macros, which are needed in several groups, are defined only once, like the ones in *Expressions*. For example, every other group, which relates to *Expressions*, contains a group-link to the group *Expressions*.

8.2.3 Definition of an *Asbru* Link

To enable the support of links, a new element, `<link>`, has been introduced to *Asbru*. It has a unique ID to identify it. Additionally, it has an attribute where the linked text of the HTML file is stored and an optional description.

It should be possible to create links to all *Asbru* elements. To spare myself the changing of every *Asbru* element, the new link-element is defined as a child of the `<comment>` - Element, which may appear nearly everywhere in an *Asbru* guideline. (As we do not need *comment*'s attribute *text*, it will always be initialized with "link".)

Therefore, when creating a link to a specific *Asbru* element, the following structure will be inserted into that element:

```
<comment text="link">
    <link id="0" linked-text="...text..."/>
</comment>
```

This implies that the link definition part of the *Asbru* macros file looks as follows:

```
<link-def>
    <element-ref name="comment">
        <element-ref name="link">
            <attribute-ref name="id" value="#"/>
            <attribute-ref name="linked-text" value="$"/>
        </element-ref>
        <attribute-ref name="text" value="link"/>
    </element-ref>
</link-def>
```

8.2.4 Sample Macros

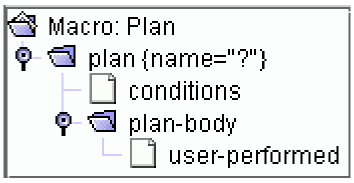
Describing all 92 macros of the *Asbru* macros file would go beyond the scope of this thesis. Therefore, four representative sample macros have been selected and will be presented hereafter. Besides a short description, the macro source code, a figure of GMT's macro preview and the corresponding *Asbru* elements are provided for each sample macro.

➢ Macro ***Plan***: This is one of the start-up macros. It can be used to insert a new plan together with the most important plan accessories. The name of the plan will be asked from the user. It is located in the 'Plans' group.

Macros File

```
<macro name="Plan">
    <element-ref name="plan">
        <attribute-ref name="name" value="?"/>
        <element-ref name="conditions"/>
        <element-ref name="plan-body">
            <element-ref name="user-performed"/>
        </element-ref>
    </element-ref>
</macro>
```

Result



```
Macro: Plan
  plan {name="?"}
    conditions
    plan-body
      user-performed
```

```
<plan name="?">
    <conditions/>
    <plan-body>
        <user-performed/>
    </plan-body>
</plan>
```

➢ Macro ***Subplans_plan-activation***: This is an example of a composed macro combined of two other macros. It provides a sub-plan together with two plan-activation elements, whereas the type of the sub-plan (which defaults to "sequentially") and the names of the referenced plans, are entered by the user. It is located in 'Plans' / 'Plan-body' / 'Subplans'.

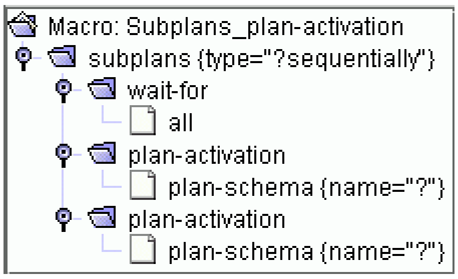
Macros File

```
<macro name="Subplans_plan-activation">
    <element-ref name="subplans">
        <attribute-ref
            name="type" value="?sequentially"/>
        <macro-ref name="Wait-For_all"/>
        <macro-ref name="Plan-Activation"/>
        <macro-ref name="Plan-Activation"/>
    </element-ref>
</macro>

<macro name="Wait-For_all">
    <element-ref name="wait-for">
        <element-ref name="all"/>
    </element-ref>
</macro>

<macro name="Plan-Activation">
    <element-ref name="plan-activation">
        <element-ref name="plan-schema">
            <attribute-ref
                name="name" value="?"/>
        </element-ref>
    </element-ref>
</macro>
```

Result

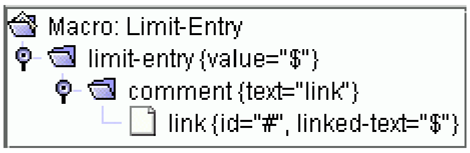


```
Macro: Subplans_plan-activation
  subplans {type="?sequentially"}
    wait-for
      all
    plan-activation
      plan-schema {name="?"}
    plan-activation
      plan-schema {name="?"}
```

```
<subplans type="?sequentially">
    <wait-for>
        <all/>
    </wait-for>
    <plan-activation>
        <plan-schema name="?"/>
    </plan-activation>
    <plan-activation>
        <plan-schema name="?"/>
    </plan-activation>
</subplans>
```

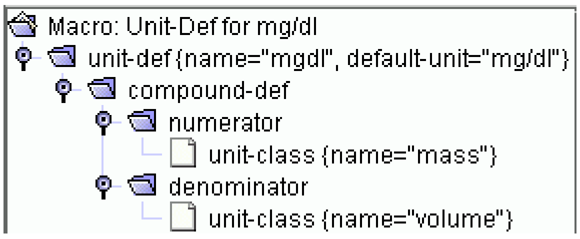
- Macro ***Limit-Entry***: This is one of the few macros, which contain a link. It inserts a limit-entry element that takes the current selection of the HTML-View as a value for its attribute 'value' and at the same time creates a link to that selection. It is located in 'Domains' / 'Parameter Definitions' / 'Qualitative Parameter Definition'.

Macros File	*Result*

`<macro name="Limit-Entry">`
`    <element-ref name="limit-entry">`
`        <attribute-ref name="value" value="$"/>`
`        <link-location/>`
`    </element-ref>`
`</macro>`
	
	Macro: Limit-Entry
	limit-entry {value="$"}
	comment {text="link"}
	link {id="#", linked-text="$"}
	`<limit-entry value="$">`
	`    <comment text="link">`
	`        <link id="#"`
	`              linked-text="$"/>`
	`    </comment>`
	`</limit-entry>`

- Macro ***Unit-Def_mgdl***: This is a macro with pre-defined attribute values and without user-input. It can be used to insert the unit type definition for 'milligram per deciliter'. It is located in 'Definitions' / 'predefined Definitions'.

Macros File	*Result*

`<macro name="Unit-Def_mgdl">`
`   <element-ref name="unit-def">`
`      <attribute-ref`
`            name="name" value="mgdl"/>`
`      <attribute-ref`
`            name="default-unit"`
`            value="mg/dl"/>`
`      <element-ref name="compound-def">`
`         <element-ref name="numerator">`
`            <element-ref name="unit-class">`
`               <attribute-ref`
`                     name="name"`
`                     value="mass"/>`
`            </element-ref>`
`         </element-ref>`
`         <element-ref name="denominator">`
`            <element-ref name="unit-class">`
`               <attribute-ref`
`                     name="name"`
`                     value="volume"/>`
`            </element-ref>`
`         </element-ref>`
`      </element-ref>`
`   </element-ref>`
`</macro>`
	
	Macro: Unit-Def for mg/dl
	unit-def {name="mgdl", default-unit="mg/dl"}
	compound-def
	numerator
	unit-class {name="mass"}
	denominator
	unit-class {name="volume"}
	`<unit-def name="mgdl"`
	`        default-unit="mg/dl">`
	`    <compound-def>`
	`        <numerator>`
	`            <unit-class`
	`                name="mass"/>`
	`        </numerator>`
	`        <denominator>`
	`            <unit-class`
	`                name="volume"/>`
	`        </denominator>`
	`    </compound-def>`
	`</unit-def>`

III Evaluation and Conclusions

9 Evaluation

To assure the usefulness of the *Guideline Markup Tool* and the *Asbru* Macros presented in this thesis, a small user study was performed. The main purpose of this study was to get feedback about the usefulness of the GMT and the *Asbru* macros. We were interested in the opinion of persons, who are not involved in this project.

Additionally, we hoped to get some suggestions about which features are missing in the GMT. This would help to improve the GMT to a more sophisticated application, which stands comparison with other professional tools.

This chapter gives some facts about the participants in the study, describes the procedure of an evaluation session and presents and discusses the results.

There was another great source of feedback for me, which should be mentioned here, although it did not belong to the 'official' evaluation phase. While I was still working on the final release of the GMT, a Dutch student used (or, to be precise, evaluated) the GMT already for her master's thesis ([Geldof, 2002]). Besides finding some bugs, she gave me many helpful recommendations for improvements.

9.1 Sample

The sample size (6 male, 2 female persons) does not look significant, especially compared with other polls. Besides the fact that I did not have the time and the capabilities to consult more persons, the rationale of the small sample size is that the GMT is intended for a relatively small and specific subset of users. A GMT user should be an expert in the domain of clinical protocols and should know *Asbru* well enough to start creating a new *Asbru* plan – these conditions reduces the number of possible users considerably.

The selection of the test persons for this study reflects these facts only partially. As far as I know, there are currently only two or three persons, where these conditions apply completely, and which are not involved into this project. Therefore, the primary selection criterion for the sample was that each person – at least – knows *Asbru*, which should imply that he/she has basic knowledge of clinical protocols.

Five of the test persons are or were involved in the development of *Asbru*. The others are rather new to the *Asgaard*/*Asbru* project.

Only one of the participating persons is a practicing physician; four of the others work at the university, one is employed in the private industry and one is serving the alternative civilian service.

The evaluation of the forms, which were filled in by the participants at the beginning of each session (see Figure A.1, p. 47, for a complete list of the questions), leads to the following additional facts about the sample:

- Seven of the participants rate their computer skills with "Professional", only one with "Intermediate" and none with "Beginner".
- All participants use a computer at work and at home.
- All of the participants are familiar with word-processing software, spreadsheet software, file managers and internet browsers. Six of the participants were also familiar with an XML editor.
- All but one participant use a programming language – the specified languages are (in order of their occurrences): Java, C/C++, Perl, PHP, Pascal and SmallTalk.
- Five participants state that they are familiar with medical terminology and clinical protocols (at least as far as it is needed for the evaluation).
- All participants specified that they know XML and *Asbru*.

9.2 Evaluation Session

Each participant was consulted separately. At first, the GMT was introduced and the participant was informed about the purpose of the evaluation session. Then the participant had to fill in a short questionnaire about her/his skills (Figure A.3, p. 49). The filled-in form is used to adapt the following explanation of the GMT to the needs of the participant.

After that, all features of the GMT were demonstrated and explained to the participant. Afterwards, the participant was asked to give the GMT a trial and test some of the features, primarily to create a new *Asbru* file, to insert one or more macros and to create an HTML-XML link.

Finally, the participant had to fill in a second form with questions about her/his impressions of the GMT and the Asbru macros (see Figure A.2, p. 48).

Each evaluation session lasted between 45 minutes and one and a half hour.

Two different computer systems were used for the evaluation: a PC (Pentium II, 333 MHz, 128 MB, Windows 2000, Java 1.4.1) at the Institute of Software Technology and a Notebook (Pentium, 150 MHz, 48MB RAM, Windows 95, Java 1.4.0). I have tried to use the first one as far as it was possible, because of its better performance. Two evaluation sessions could not take place at the institute, so the second one has to be used, too.

9.3 Results

The evaluation of the feedback forms, which were filled in by the participants at the end of each session (see Figure A.4, p. 50, for a complete list of the questions), leads to the following results.

9.3.1 Overall Impression

All participants have a "Good" overall impression of the system. Two of them even complained that there was no "Very Good" option on the form.

Seven participants rated the speed of the program with "Good", only one with "Medium". Six participants judged the handling of the program with "Good" and two with "Medium".

Nobody thought that the *GMT* is not useable in practice.

9.3.2 XML View

Seven rated the functionality of the XML view with "Good", only one with "Medium".

Five participants thought that the functionality of the XML view is sufficient. The others recommended following additional features: copy/cut&paste and drag'n'drop of XML nodes, undo, replace instead of insert elements/macros and the ability to get a list of elements, which are insertable into the current XML node.

9.3.3 Macros Structure View

All participants rated the handling of the macros view with "Good". Nobody found the representation of the hierarchical macros structure with list boxes unclear.

Seven participants thought that the functionality of the macros view is sufficient (one abstention). Recommendations regarding the macros view include context-sensitive help (about macros), editable macros and the ability to see where the current macro could be inserted in the XML file.

9.3.4 Asbru Macros

Seven participants thought that the structuring of the macros is clear (one abstention).

Five participants believed that the structure of the macros increases the insight into *Asbru*, whereas two participants disbelieved this (one abstention).

Five participants thought that the amount of macros is sufficient – only one participant disagreed (two abstentions).

Six participants found the meaning of the macros obvious (two abstentions).

All participants believed that the macros would support the workflow.

9.4 Conclusions of the Evaluation

Altogether, the results approve that we are going the right way. All participants were impressed by the ideas and concepts of the *Guideline Markup Tool*. They all comprehend the advantages, which were involved by the possibility to link the XML file with the original guideline.

Everyone quickly understood the usage of the program after a short demonstration. They were all able to try out the main features for themselves.

Especially the participants, who have no experience with *Asbru*, complained that it is difficult to know which macro is best suited in a particular situation. This has to be considered in the future development of the GMT. One suggestion regarding this point was to see all eligible macros for the currently selected element and, the other way round, to highlight all matching elements in the XML view regarding the currently selected macro.

The evaluation has shown that there are still several features missing. Many participants told me their ideas of possible improvements. The most wanted improvement was certainly the possibility to undo the last action. Since all participants were completely new to the GMT, this feature was often demanded while they were trying out the GMT.

All recommendations from the participants during the evaluation sessions regarding improvements of the GMT and the macros file are included in chapter 11, "Future Work".

10 Summary

In the medical domain, clinical guidelines have been arisen to unify treatment of diseases. Guidelines tend to have some shortcomings, which made them not very practicable. Several efforts have been undertaken to improve the usefulness of guidelines, especially to prove their correctness. This has been done by trying to model guidelines in a machine-readable form. Eight of these guideline representation methods have been outlined in this thesis. Additionally, some guideline tools have been presented.

In this thesis, a new guideline tool, called *Guideline Markup Tool* (GMT), has been introduced. It aims to help knowledge engineers in translating clinical guidelines into their *Asbru* representation. It does this by providing 'macros' to facilitate the assembling of *Asbru* guidelines. A macro contains a structure of *Asbru* elements, which belong together.

But the main feature of the GMT is the ability to create and maintain links between a guideline HTML file and its representing *Asbru* XML file. The knowledge engineer should always define links during the translation task, at least for every important *Asbru* element pointing to the corresponding part in the original guideline. If the resulted *Asbru* XML file is used as an input of another *Asgaard* tool, it may happen that someone wants to know the reason for the choice of a particular *Asbru* element or the origin of a specific attribute value. The GMT can be used to answer such questions, by clicking on the nearest link, which will highlight the corresponding part in the HTML file.

To be consequently, links also work in the other direction. This allows easier comprehension of the translation process and thereby facilitates learning of the quite complex language *Asbru*. Therewith, the GMT can be used to find out how a particular passage in the text of the original guideline has been modelled in *Asbru*.

To demonstrate the applicability of the abovementioned macros and to increase the usefulness of the GMT, an *Asbru* macros file has been developed as an additional goal of this thesis. The main focus of the development of the macros file was a sophisticated structuring of the available macros, which should make finding a proper macro easier.

11 Future Work

This chapter should show that there is still much room for improvement concerning the GMT and the *Asbru* macros file. The evaluation showed that the current version is ready to be used in daily work, but that there are many things, which could be improved.

The projected future work will be presented in two parts: a list of improvements regarding the GMT and a second one concerning the *Asbru* macros file.

11.1 Improvements of the Guideline Markup Tool

Since I get so many suggestions regarding the GMT, especially during the evaluation phase, I have grouped them by their priority, whereas "high priority" means that it is important to enable daily work with the GMT, "medium priority" indicates that it would increase user convenience and "low priority" denotes nice, but not really necessary extras.

If you are interested in the progress of the Guideline Markup Tool, keep an eye on the status page of the GMT website ([URL #43]).

11.1.1 High Priority

- Facilitate the choice of proper macros through listing all macros (in an extra view), which were allowed to be inserted into the current element. An additional feature should mark all elements in the XML-View, where a specific macro could be inserted.
- Project: Since an *Asbru* XML file belongs always to one HTML file, it would be comfortable to put them together with one (or more) macros files into a "project", so the user could open all needed files at once.
- Undo / Redo: Because it is always possible, that a user removes an element (with all its children) by mistake, it should be able to undo the previously action. Of course, it should also be able to revoke 'undo' (redo).
- Linking to / from images in HTML-View: Guidelines, written in HTML, often use images to illustrate the workflow of a treatment. Therefore, the GMT would be much more useful, if it allows the creation of links inside of images. The problem thereby lies not in the internal definition of such image links (because this is feasible in HTML), but in providing the right user handling.
- Nested Links in HTML: If a link is defined to an entire paragraph in the HTML file, it should be possible to define another link to a word or sentence inside this paragraph.
- Search / Replace in XML view: Especially when editing a large XML file, it would be useful to be able to search for a specific element or attribute value and replace all occurrences of a value.

11.1.2 Medium Priority

- Attributed Links: It should be possible to define arbitrary attributes to a link, to divide links into different categories. This could also be used to specify if it is an *explicit* or an *implicit* Link, as proposed by [Geldof, 2002].
- Copy / Cut / Paste in XML-View: Most users are used to the *clipboard* feature, which is integrated in many applications. This could be applied to duplicate or move elements in the XML-View.
- Resize (Move) Links in HTML-View: It should be possible to change the extent of a link in the HTML-View, in hindsight, without the need to remove and re-create it.
- Help about Macros: Especially people, who are new to *Asbru*, could need more detailed information about the meaning of the macros and elements than provided in a macros file. Therefore, it would be useful to point at a specific entry of an external help resource (like the *Asbru* documentation) related to the current macro.
- Insert Above / Insert Below / Replace: In addition to inserting a macro into an element, it should be possible to insert it above or below an element or even replace the current element.

- Search in HTML-View: Like searching in the XML-View, it may be also useful to locate a given keyword (for example 'treatment') in the HTML document.
- Editing of Macros File: A user might want to have the potentiality to adapt a macro or even the entire macros file in a more comfortable way than using a text editor. It has to be decided, if this feature will be integrated into the GMT or better realised as an external application.
- User-defined Icons in XML-View: It should be possible for the user, to specify different icons for specific XML elements. This would facilitate locating of important elements.

11.1.3 Low Priority

- Insert Elements without Macros File: At present, single elements have to be also modeled in the macros file. Instead, an extra list of elements, which are allowed to be inserted into the current element, could be shown.
- Simple Editing in HTML-View: If an XML element relates to a list item (<li> tag in HTML) of an unordered list (<ul> tag) in the HTML file, it should be possible to replace the list bullet (•) with a character (like '*') which can be selected and used as a target of a link. But it is in no case intended to provide full HTML editing features like HTML editors.
- Additional View(s) of XML-File: It may be useful, to provide different views of an XML document (eventually realized through a plug-in mechanism).
- Editing of comment / text nodes in XML view: It should be possible to change the text of a comment or even text node in the XML-View.
- Drag'n'Drop in XML-View: This is not really necessary for the usage of the GMT, but since enough people are used to use Drag'n'Drop in their daily work, it should be taken into account.
- Support for any XML language: If it is desired to use the GMT with other XML languages, the template that is used for creating a new XML file, and the internal predefinition of a link should be modifiable by the user.

11.2 Improvements of the Macros File

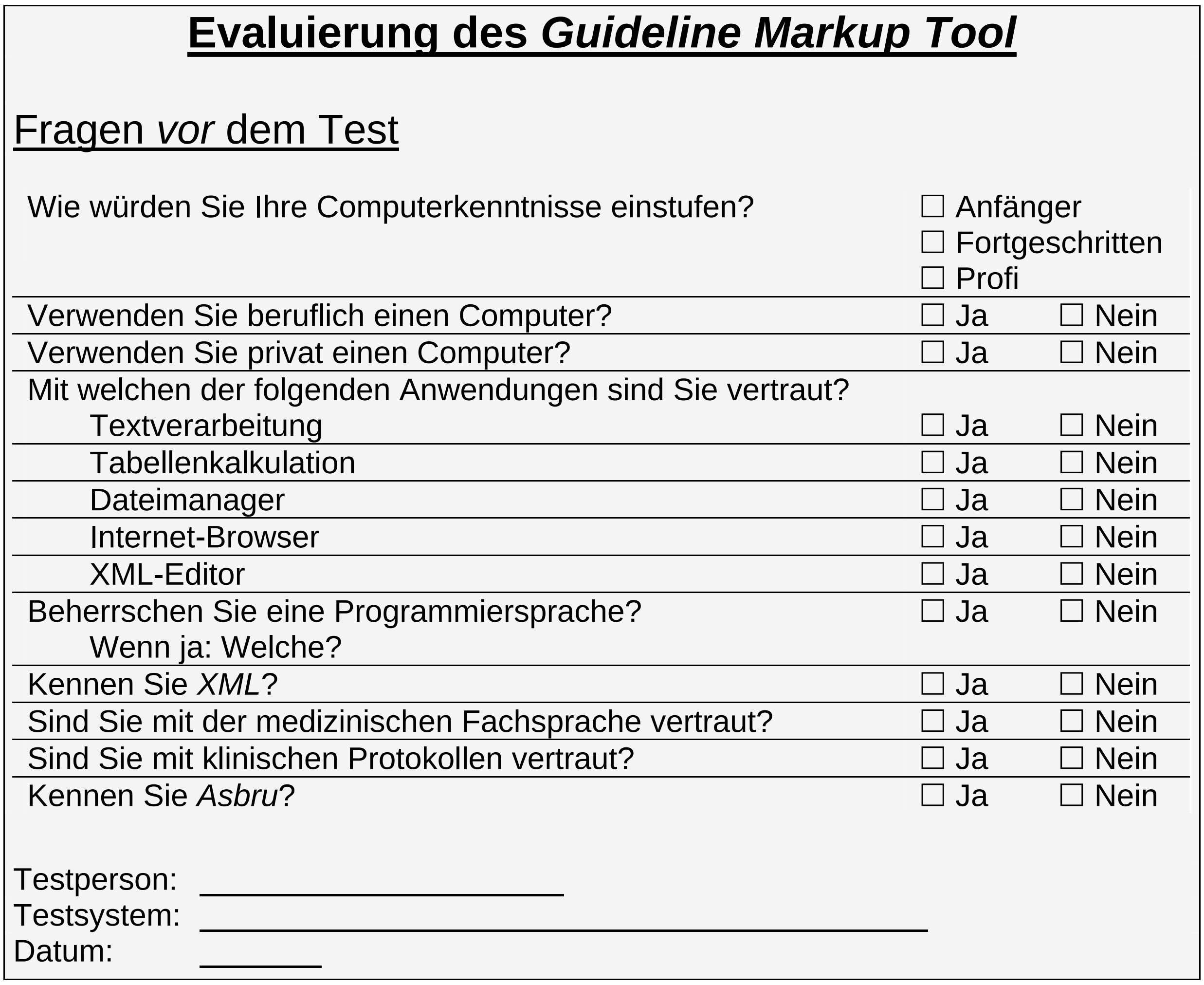
- Context dependent macros: A *context dependent macro* is a macro whose content depends on the target element. To accomplish this, the macro definition in the macros file has to be extended to provide contents for different situations. For instance, consider an *Asbru* macro 'Domain', which inserts a *domain* element including a parameter-group if applied to a *domain-defs* element. When the same macro would be applied to a *plan-library* element that has no *domain-defs* element, the *domain-defs* element would be inserted, too.
- More macros: The current *Asbru* macros file does only support a selection of all available *Asbru* elements. To achieve the final aim of a complete macros file, much more macros have to be written.
- Better descriptions of macros: Each macro should include a more meaningful description, which should help the user choosing the proper macro for a particular situation.

Appendix

A Questionnaires

This chapter presents the questionnaires used during the evaluation phase and their English translations. Ideas for the questionnaires were taken from [Kosara, 1999].

A.1 German Version: Original Questionnaire



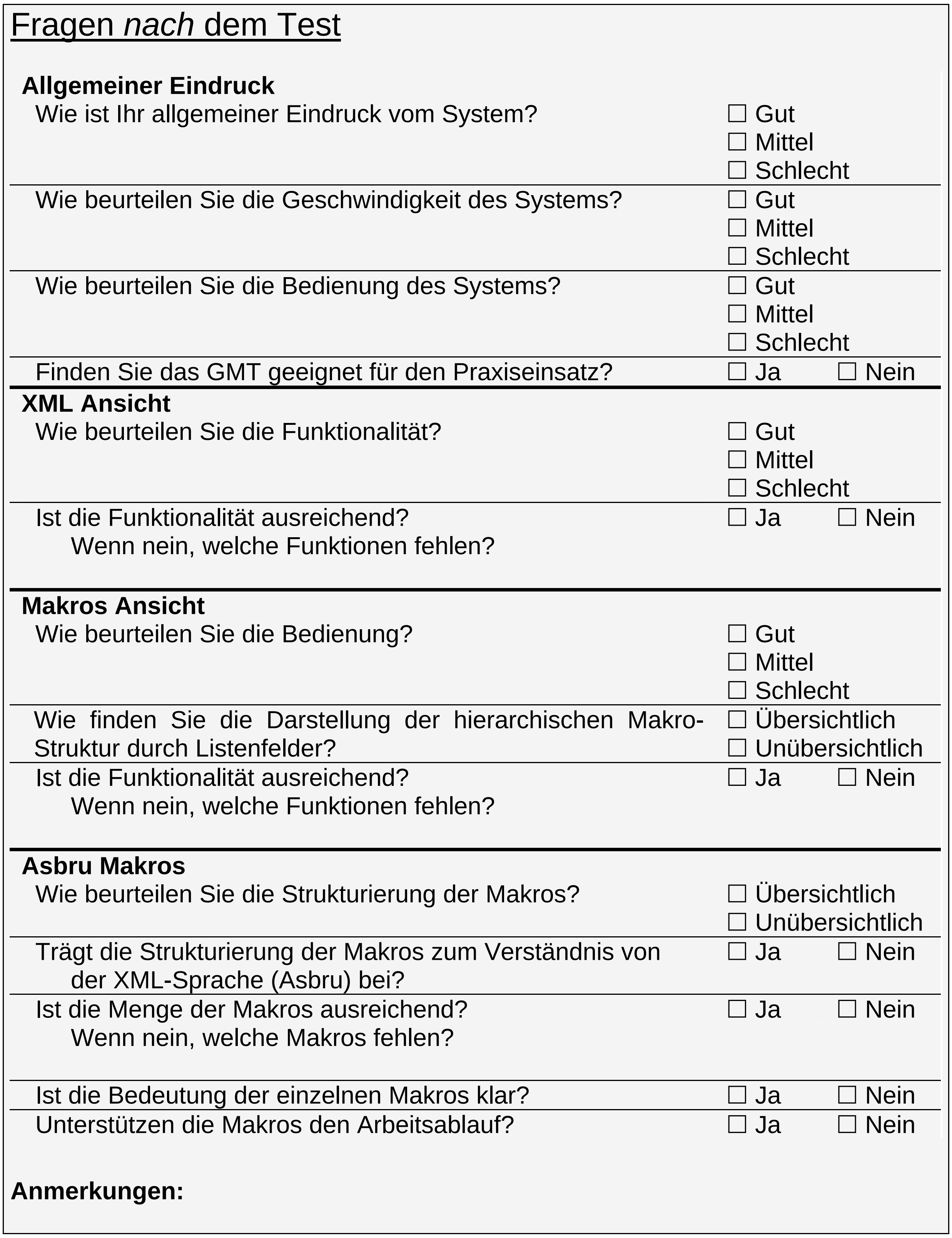
Evaluierung des *Guideline Markup Tool*

Fragen *vor* dem Test

Wie würden Sie Ihre Computerkenntnisse einstufen?	☐ Anfänger ☐ Fortgeschritten ☐ Profi	
Verwenden Sie beruflich einen Computer?	☐ Ja	☐ Nein
Verwenden Sie privat einen Computer?	☐ Ja	☐ Nein
Mit welchen der folgenden Anwendungen sind Sie vertraut? Textverarbeitung	☐ Ja	☐ Nein
Tabellenkalkulation	☐ Ja	☐ Nein
Dateimanager	☐ Ja	☐ Nein
Internet-Browser	☐ Ja	☐ Nein
XML-Editor	☐ Ja	☐ Nein
Beherrschen Sie eine Programmiersprache? Wenn ja: Welche?	☐ Ja	☐ Nein
Kennen Sie *XML*?	☐ Ja	☐ Nein
Sind Sie mit der medizinischen Fachsprache vertraut?	☐ Ja	☐ Nein
Sind Sie mit klinischen Protokollen vertraut?	☐ Ja	☐ Nein
Kennen Sie *Asbru*?	☐ Ja	☐ Nein

Testperson: ______________________
Testsystem: ____
Datum: _______

Figure A.1: Questions before Test (German version)



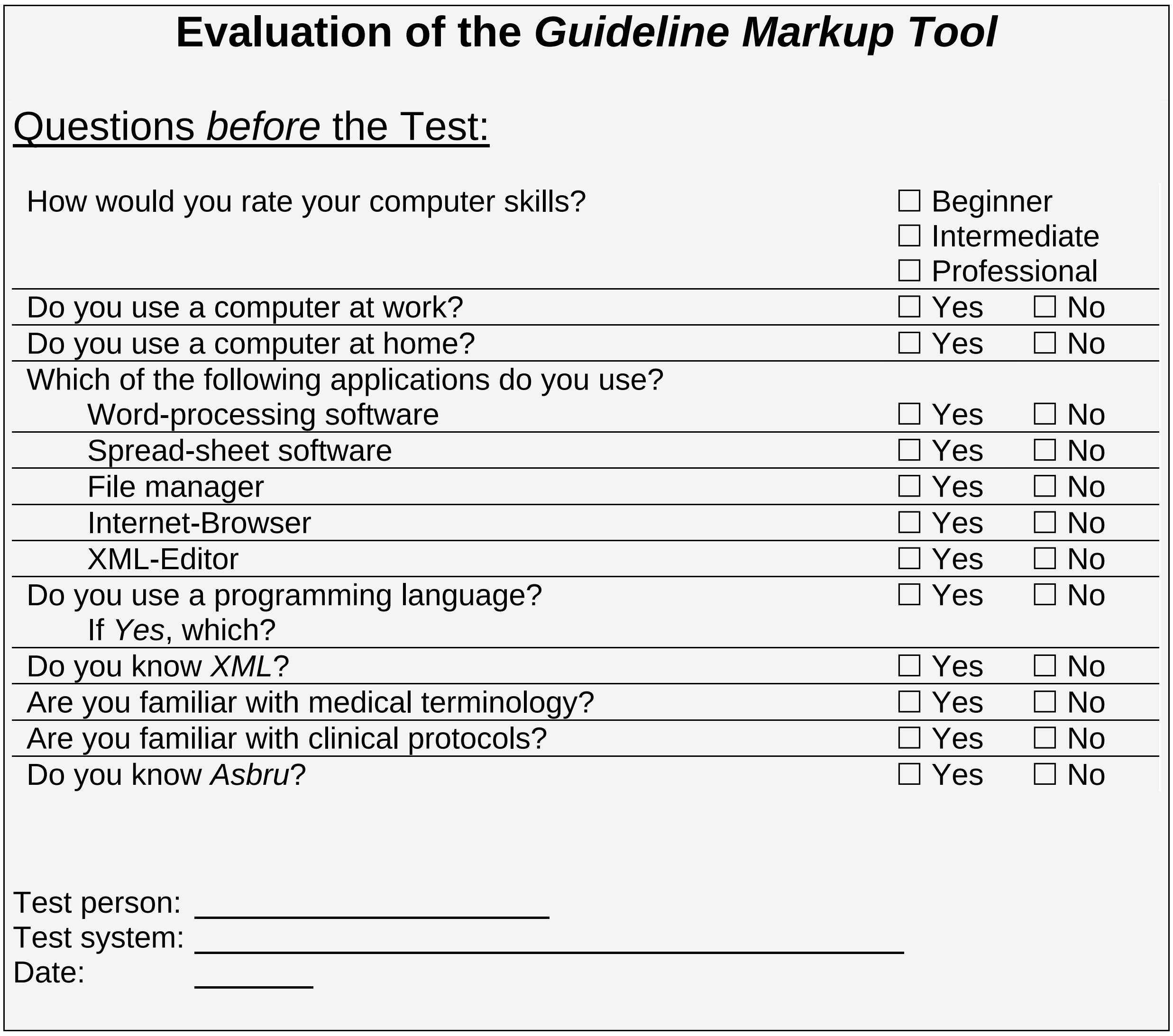
Fragen *nach* dem Test

Allgemeiner Eindruck	
Wie ist Ihr allgemeiner Eindruck vom System?	☐ Gut ☐ Mittel ☐ Schlecht
Wie beurteilen Sie die Geschwindigkeit des Systems?	☐ Gut ☐ Mittel ☐ Schlecht
Wie beurteilen Sie die Bedienung des Systems?	☐ Gut ☐ Mittel ☐ Schlecht
Finden Sie das GMT geeignet für den Praxiseinsatz?	☐ Ja ☐ Nein
XML Ansicht	
Wie beurteilen Sie die Funktionalität?	☐ Gut ☐ Mittel ☐ Schlecht
Ist die Funktionalität ausreichend? Wenn nein, welche Funktionen fehlen?	☐ Ja ☐ Nein
Makros Ansicht	
Wie beurteilen Sie die Bedienung?	☐ Gut ☐ Mittel ☐ Schlecht
Wie finden Sie die Darstellung der hierarchischen Makro-Struktur durch Listenfelder?	☐ Übersichtlich ☐ Unübersichtlich
Ist die Funktionalität ausreichend? Wenn nein, welche Funktionen fehlen?	☐ Ja ☐ Nein
Asbru Makros	
Wie beurteilen Sie die Strukturierung der Makros?	☐ Übersichtlich ☐ Unübersichtlich
Trägt die Strukturierung der Makros zum Verständnis von der XML-Sprache (Asbru) bei?	☐ Ja ☐ Nein
Ist die Menge der Makros ausreichend? Wenn nein, welche Makros fehlen?	☐ Ja ☐ Nein
Ist die Bedeutung der einzelnen Makros klar?	☐ Ja ☐ Nein
Unterstützen die Makros den Arbeitsablauf?	☐ Ja ☐ Nein

Anmerkungen:

Figure A.2: Questions after Test (German version)

A.2 English version of the Questionnaire



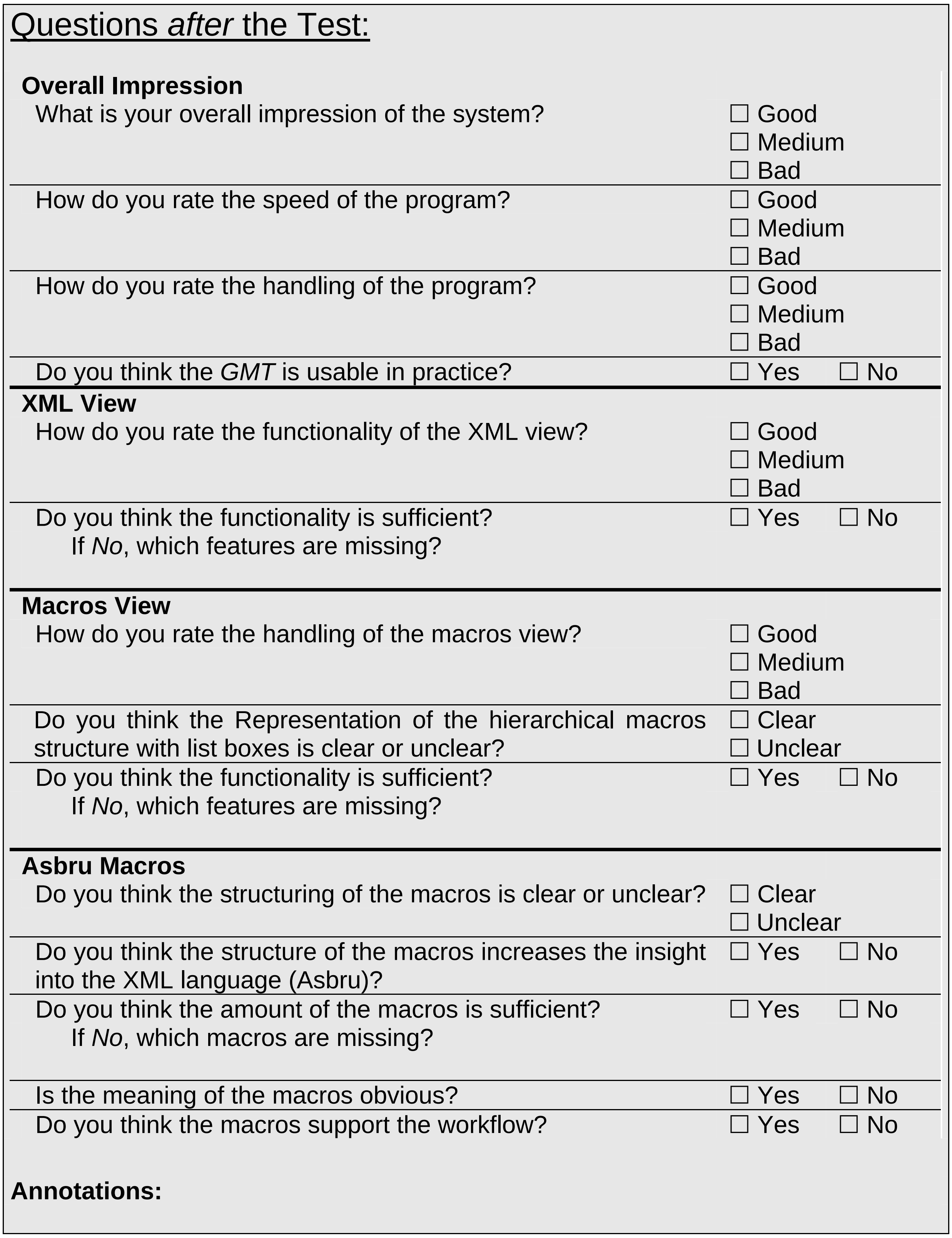
Evaluation of the *Guideline Markup Tool*

<u>Questions *before* the Test:</u>

How would you rate your computer skills?	☐ Beginner ☐ Intermediate ☐ Professional	
Do you use a computer at work?	☐ Yes	☐ No
Do you use a computer at home?	☐ Yes	☐ No
Which of the following applications do you use? Word-processing software	☐ Yes	☐ No
Spread-sheet software	☐ Yes	☐ No
File manager	☐ Yes	☐ No
Internet-Browser	☐ Yes	☐ No
XML-Editor	☐ Yes	☐ No
Do you use a programming language? If *Yes*, which?	☐ Yes	☐ No
Do you know *XML*?	☐ Yes	☐ No
Are you familiar with medical terminology?	☐ Yes	☐ No
Are you familiar with clinical protocols?	☐ Yes	☐ No
Do you know *Asbru*?	☐ Yes	☐ No

Test person: ______________________
Test system: ____
Date: _______

Figure A.3: Questions before test (English version)



Questions *after* the Test:

Overall Impression	
What is your overall impression of the system?	☐ Good ☐ Medium ☐ Bad
How do you rate the speed of the program?	☐ Good ☐ Medium ☐ Bad
How do you rate the handling of the program?	☐ Good ☐ Medium ☐ Bad
Do you think the *GMT* is usable in practice?	☐ Yes ☐ No
XML View	
How do you rate the functionality of the XML view?	☐ Good ☐ Medium ☐ Bad
Do you think the functionality is sufficient? If *No*, which features are missing?	☐ Yes ☐ No
Macros View	
How do you rate the handling of the macros view?	☐ Good ☐ Medium ☐ Bad
Do you think the Representation of the hierarchical macros structure with list boxes is clear or unclear?	☐ Clear ☐ Unclear
Do you think the functionality is sufficient? If *No*, which features are missing?	☐ Yes ☐ No
Asbru Macros	
Do you think the structuring of the macros is clear or unclear?	☐ Clear ☐ Unclear
Do you think the structure of the macros increases the insight into the XML language (Asbru)?	☐ Yes ☐ No
Do you think the amount of the macros is sufficient? If *No*, which macros are missing?	☐ Yes ☐ No
Is the meaning of the macros obvious?	☐ Yes ☐ No
Do you think the macros support the workflow?	☐ Yes ☐ No

Annotations:

Figure A.4: Questions after test (English version)

B Installation Instructions of the GMT

There are two possibilities to use the GMT, either using *Java Web Start (JWS)* or by manually downloading and running it local. Both need the Java Runtime Environment (JRE) since version 1.3.0 ([URL #45]).

JWS represents certainly the easiest method. With JWS always the latest version of the GMT will be used. Since Java 1.4.0, JWS is included in the JRE. For older Java versions, JWS has to be downloaded ([URL #46]) and installed before. To run GMT using JWS, click on the "Launch GMT" button on the GMT web page ([URL #37]). This will open the *Java Web Start Manager* that automatically download all required files and start the GMT.

For manually downloading, the required files can be found at the "GMT Download page" ([URL #44]). Additional to `gmt.jar,` the following libraries are needed: an XML-parser (*Xerces* [URL #38] or *Crimson* [URL #39]), a XSL Stylesheet Processor (*Xalan* [URL #40] or Saxon [URL #41]) and *XML4J*. To get them all together, they were available as .zip or .tar.gz packages. After downloading, the GMT can be started either by double-clicking on `gmt.jar`, by executing the scripts (`gmt.bat` or `gmt.sh`) or by entering `java -jar gmt.jar` in the command line.

C The Macros XML Language

This appendix presents an overview of all possible elements of the macros XML language. Furthermore, the source code of the appertaining document type definition is included (a commented version of the DTD file and an automatically generated HTML documentation can be found at [URL #42]).

element	**occurs in**	**may contain**	**icon in GMT**
`gmt`	– (root-element)	`link-def, macros, group`	–
`link-def`	`gmt`	`element-ref`	–
`macros`	`gmt`	`macro-group, macro`	–
`macro-group`	`macros, macro-group`	`macro-group, macro`	–
`macro`	`macros, macro-group`	`element-ref, macro-ref, link-location`	–
`element-ref`	`link-def, macro, element-ref`	`attribute-ref, element-ref, macro-ref, link-location`	–
`attribute-ref`	`element-ref`	–	–
`link-location`	`macro, element-ref`	–	–
`group`	`gmt, group`	`group, group-ref, group-link, macro-ref, single-element-ref, separator, label`	📁, 📂
`macro-ref`	`macro, element-ref, group`	–	📄, 📄, 🔗, 🔗 (in group)
`group-ref`	`group`	–	see group
`group-link`	`group`	–	➘
`single-element-ref`	`group`	–	📄, 📄
`separator`	`group`	–	line
`label`	`group`	-	text

Figure C.1: Overview of elements of a macros file

```
<!ELEMENT gmt (link-def, macros, group)>

<!ELEMENT link-def (element-ref)>
<!ELEMENT macros ((macro-group | macro)+)>
<!ELEMENT macro-group ((macro-group | macro)+)>
<!ATTLIST macro-group
    description CDATA #REQUIRED
>
<!ELEMENT macro (element-ref | macro-ref | link-location)+>
<!ATTLIST macro
    name ID #REQUIRED
    description CDATA #IMPLIED
    standalone (true | false) "true"
>
<!ELEMENT element-ref (attribute-ref | element-ref | macro-ref | link-location)*>
<!ATTLIST element-ref
    name NMTOKEN #REQUIRED
>
<!ELEMENT attribute-ref EMPTY>
<!ATTLIST attribute-ref
    name NMTOKEN #REQUIRED
    value CDATA #REQUIRED
    default-value CDATA #IMPLIED
>
<!ELEMENT link-location EMPTY>
<!ELEMENT group (group | group-ref | group-link | macro-ref | single-element-ref |
separator | label)*>
<!ATTLIST group
    name CDATA #REQUIRED
    description CDATA #IMPLIED
>
<!ELEMENT macro-ref EMPTY>
<!ATTLIST macro-ref
    name IDREF #REQUIRED
    label CDATA #IMPLIED
>
<!ELEMENT group-ref EMPTY>
<!ATTLIST group-ref
    name CDATA #REQUIRED
    label CDATA #IMPLIED
>
<!ELEMENT group-link EMPTY>
<!ATTLIST group-link
    name CDATA #REQUIRED
    label CDATA #IMPLIED
    description CDATA #IMPLIED
>
<!ELEMENT single-element-ref EMPTY>
<!ATTLIST single-element-ref
    name NMTOKEN #REQUIRED
    description CDATA #IMPLIED
    standalone (true | false) "true"
>
<!ELEMENT separator EMPTY>
<!ELEMENT label EMPTY>
<!ATTLIST label
    text CDATA #REQUIRED
>
```

Figure C.2: Source Code of `gmt.dtd`

D Excerpts from the Asbru Macros File

Since the entire Asbru macros file would take more than 15 pages, only excerpts are presented here. To increase the legibility, the *description* and *label* attributes have been omitted. The entire Asbru macros file, together with a syntax-coloured HTML version and a better readable HTML overview, can be found at [URL #42].

```
<?xml version="1.0"?>
<!DOCTYPE gmt SYSTEM "gmt.dtd">
<gmt>
  <link-def>
    <element-ref name="comment">
      <element-ref name="link">
        <attribute-ref name="id" value="#"/>
        <attribute-ref name="linked-text" value="$"/>
      </element-ref>
      <attribute-ref name="text" value="link"/>
    </element-ref>
  </link-def>
  <macros>
    <macro-group description="Plan macros">
      <macro name="Plan">
        <element-ref name="plan">
          <attribute-ref name="name" value="?"/>
          <element-ref name="conditions"/>
          <element-ref name="plan-body">
            <element-ref name="user-performed"/>
          </element-ref>
        </element-ref>
      </macro>
      <macro name="Intentions" standalone="false">
        <element-ref name="intentions">
          <macro-ref name="Intention"/>
        </element-ref>
      </macro>
      <macro name="Intention" standalone="false">
        <element-ref name="intention">
          <attribute-ref name="type" value="?"/>
          <attribute-ref name="verb" value="?"/>
        </element-ref>
      </macro>
      <macro name="Refer-To">
        <element-ref name="refer-to">
          <attribute-ref name="plan-name" value="?"/>
          <attribute-ref name="label" value="?"/>
        </element-ref>
      </macro>
      <macro name="Abort-Condition_with_Comparison" standalone="false">
        <element-ref name="abort-condition">
          <macro-ref name="Simple-Condition_Comparison"/>
        </element-ref>
      </macro>
      <macro name="Filter-Precondition_confirmation-required">
        <element-ref name="filter-precondition">
          <attribute-ref name="confirmation-required" value="yes"/>
          <element-ref name="none"/>
        </element-ref>
      </macro>
      <macro-group description="Plan-Body">
        <macro name="Plan-Body_user-performed">
          <element-ref name="plan-body">
            <element-ref name="user-performed"/>
          </element-ref>
        </macro>
        <macro name="Plan-Body_plan-activation">
          <element-ref name="plan-body">
            <element-ref name="plan-activation">
              <element-ref name="plan-schema">
                <attribute-ref name="name" value="?"/>
              </element-ref>
            </element-ref>
          </element-ref>
        </macro>
        <macro name="Plan-Body_subplans_plan-activation">
          <element-ref name="plan-body">
            <macro-ref name="Subplans_plan-activation"/>
          </element-ref>
        </macro>
        <macro name="Plan-Body_Cyclical-Plan">
          <element-ref name="plan-body">
            <macro-ref name="Cyclical-Plan"/>
          </element-ref>
        </macro>
        <macro-group description="subplans">
          [...]
        </macro-group>
        <macro-group description="plan-body_stuff">
          [...]
        </macro-group>
      </macro-group>
      <macro-group description="Plan Stuff">
        <macro-group description="temporal patterns">
          [...]
        </macro-group>
        <macro name="Plan-Group">
          <element-ref name="plan-group">
            <attribute-ref name="title" value="?"/>
          </element-ref>
        </macro>
        <macro name="Derived-From">
          <element-ref name="derived-from">
            <attribute-ref name="plan-name" value="?"/>
          </element-ref>
        </macro>
        <macro name="Refers-To">
          <element-ref name="refers-to">
            <attribute-ref name="plan-name" value="?"/>
          </element-ref>
        </macro>
        <macro name="Explanation">
          <element-ref name="explanation">
            <attribute-ref name="text" value="?"/>
          </element-ref>
        </macro>
      </macro-group>
    </macro-group>
    <macro-group description="domain macros">
      <macro name="Domain">
        <element-ref name="domain">
          <attribute-ref name="name" value="?"/>
        </element-ref>
      </macro>
      <macro name="Use-Domain">
        <element-ref name="use-domain">
          <attribute-ref name="name" value="?"/>
        </element-ref>
      </macro>
      <macro name="Domain-Definition">
        <element-ref name="domain-defs">
          <element-ref name="domain">
            <attribute-ref name="name" value="?"/>
```

```
            <macro-ref name="Unit-Def"/>
            <macro-ref name="Qualitative-Scale-Def"/>
            <macro-ref name="Patient-Record-Def"/>
            <macro-ref name="Parameter-Group"/>
            <macro-ref name="Variable-Def"/>
            <macro-ref name="Constant-Def"/>
          </element-ref>
        </element-ref>
      </macro>
      <macro-group description="Parameter Definitions">
        <macro name="Parameter-Group" standalone="false">
          <element-ref name="parameter-group">
            <attribute-ref name="title" value="?"/>
            <macro-ref name="Parameter-Def"/>
          </element-ref>
        </macro>
        <macro name="Parameter-Def" standalone="false">
          <element-ref name="parameter-def">
            <attribute-ref name="name" value="?"/>
            <attribute-ref name="type" value="?"/>
          </element-ref>
        </macro>
        <macro-group description="Raw Data Definitions">
          [...]
        </macro-group>
        <macro-group description="Qualitative Parameter Definitions">
          [...]
        </macro-group>
        <macro name="Change-Def">
          <element-ref name="change-def">
            <element-ref name="interval">
              <element-ref name="numerical-constant">
                <attribute-ref name="scale" value="?"/>
                <attribute-ref name="value" value="?"/>
              </element-ref>
            </element-ref>
            <element-ref name="parameter-ref">
              <attribute-ref name="name" value="?"/>
            </element-ref>
          </element-ref>
        </macro>
      </macro-group>
    </macro-group>
    <macro-group description="Definitions for Plan, Library and Domain">
      <macro name="Record-Def">
        <element-ref name="record-def">
          <macro-ref name="Field-Def"/>
        </element-ref>
      </macro>
      <macro name="Field-Def">
        <element-ref name="field-def">
          <attribute-ref name="name" value="?"/>
          <element-ref name="scalar-def">
            <attribute-ref name="type" value="?"/>
          </element-ref>
        </element-ref>
      </macro>
      <macro name="Patient-Record-Def">
        <element-ref name="patient-record-def">
          <attribute-ref name="name" value="?"/>
          <element-ref name="primary-key">
            <attribute-ref name="name" value="name"/>
          </element-ref>
          <macro-ref name="Record-Def"/>
        </element-ref>
      </macro>
      <macro name="Qualitative-Scale-Def">
        <element-ref name="qualitative-scale-def">
          <attribute-ref name="name" value="?"/>
          <element-ref name="qualitative-entry">
            <attribute-ref name="entry" value="?"/>
          </element-ref>
        </element-ref>
      </macro>
      <macro name="Qualitative-Entry">
        <element-ref name="qualitative-entry">
          <attribute-ref name="entry" value="?"/>
        </element-ref>
      </macro>
      <macro name="Numerical-Scale-Def">
        <element-ref name="numerical-scale-def">
          <attribute-ref name="name" value="?"/>
          <attribute-ref name="unit-class" value="?"/>
          <macro-ref name="Unit-Entry"/>
        </element-ref>
      </macro>
      <macro name="Unit-Entry">
        <element-ref name="unit-entry">
          <attribute-ref name="factor" value="?"/>
        </element-ref>
      </macro>
      <macro name="Unit-Def">
        <element-ref name="unit-def">
          <attribute-ref name="name" value="?"/>
          <attribute-ref name="default-unit" value="?"/>
          <element-ref name="compound-def">
            <element-ref name="numerator">
              <element-ref name="unit-class">
                <attribute-ref name="name" value="?"/>
              </element-ref>
            </element-ref>
            <element-ref name="denominator">
              <element-ref name="unit-class">
                <attribute-ref name="name" value="?"/>
              </element-ref>
            </element-ref>
          </element-ref>
        </element-ref>
      </macro>
      <macro-group description="predefined">
        <macro name="Unit-Def_mgdl">
          <element-ref name="unit-def">
            <attribute-ref name="name" value="mgdl"/>
            <attribute-ref name="default-unit" value="mg/dl"/>
            <element-ref name="compound-def">
              <element-ref name="numerator">
                <element-ref name="unit-class">
                  <attribute-ref name="name" value="mass"/>
                </element-ref>
              </element-ref>
              <element-ref name="denominator">
                <element-ref name="unit-class">
                  <attribute-ref name="name" value="volume"/>
                </element-ref>
              </element-ref>
            </element-ref>
          </element-ref>
        </macro>
      </macro-group>
      <macro name="Variable-Def" standalone="false">
        <element-ref name="variable-def">
          <attribute-ref name="name" value="?"/>
        </element-ref>
      </macro>
      <macro name="Constant-Def" standalone="false">
        <element-ref name="constant-def">
          <attribute-ref name="name" value="?"/>
        </element-ref>
      </macro>
      <macro name="Constant-Def_with-numerical-scalar-def" standalone="false">
        <element-ref name="constant-def">
          <attribute-ref name="name" value="?"/>
          <element-ref name="scalar-def">
            <attribute-ref name="type" value="?"/>
            <element-ref name="initial-value">
              <element-ref name="numerical-constant">
                <attribute-ref name="value" value="?" default-value="0"/>
              </element-ref>
            </element-ref>
          </element-ref>
        </element-ref>
      </macro>
      <macro name="Iterator-Def" standalone="false">
        <element-ref name="iterator-def">
          <attribute-ref name="name" value="?"/>
        </element-ref>
      </macro>
      <macro name="Iterator-Def_on-variable">
        <element-ref name="iterator-def">
          <attribute-ref name="name" value="?"/>
          <element-ref name="variable-ref">
            <attribute-ref name="name" value="?"/>
          </element-ref>
        </element-ref>
      </macro>
    </macro-group>
    <macro-group description="Pattern">
      <macro-group description="Tables">
        <macro-group description="Qualitative Parameter Definitions">
          <macro name="Param-Def-Table" standalone="false">
            <element-ref name="parameter-def">
              <element-ref name="qualitative-parameter-def">
                <element-ref name="parameter-ref">
                  <attribute-ref name="name" value="?"/>
                </element-ref>
              </element-ref>
            </element-ref>
          </macro>
          <macro name="Param-Def-Row">
            <element-ref name="limits">
              <attribute-ref name="unit" value="?"/>
              <element-ref name="context">
                <element-ref name="one-of">
                  <attribute-ref name="name" value="?"/>
                  <element-ref name="value-ref">
                    <attribute-ref name="name" value="?"/>
                  </element-ref>
                </element-ref>
              </element-ref>
            </element-ref>
          </macro>
          <macro name="Param-Def-Column">
            <element-ref name="limit-entry">
              <attribute-ref name="value" value="$"/>
              <link-location/>
            </element-ref>
          </macro>
        </macro-group>
      </macro-group>
    </macro-group>
    <macro-group description="misc">
      <macro-group description="Expressions">
        [...]
      </macro-group>
      <macro name="Library-Info">
        <element-ref name="library-info">
          <attribute-ref name="title" value="?"/>
          <attribute-ref name="version" value="1"/>
          <element-ref name="administrative-data">
            <attribute-ref name="original-author" value="?"/>
            <attribute-ref name="creation-date" value="%"/>
          </element-ref>
        </element-ref>
      </macro>
      <macro name="Changes">
        <element-ref name="changes">
          <attribute-ref name="author" value="?"/>
          <attribute-ref name="date" value="%"/>
          <attribute-ref name="description" value="?"/>
        </element-ref>
      </macro>
      <macro name="Changes_Last-update">
        <element-ref name="changes">
          <attribute-ref name="author" value="?"/>
          <attribute-ref name="date" value="%"/>
          <attribute-ref name="description" value="Date of last update"/>
        </element-ref>
      </macro>
      <macro-group description="any-comment">
        <macro name="Link">
          <link-location/>
        </macro>
        <macro name="Comment">
          <element-ref name="comment">
            <attribute-ref default-value="$" name="text" value="?"/>
          </element-ref>
        </macro>
        <macro name="URL">
          <element-ref name="url">
            <attribute-ref name="text" value="?"/>
          </element-ref>
        </macro>
        <macro name="BibRef">
          <element-ref name="bibref">
            <attribute-ref name="key" value="?"/>
          </element-ref>
        </macro>
      </macro-group>
    </macro-group>
  </macros>
  <group name="root">
    <group name="Plans">
      <macro-ref name="Plan"/>
      <group name="Conditions">
        <single-element-ref name="conditions"/>
        <separator/>
        <single-element-ref name="filter-precondition" standalone="false"/>
        <macro-ref name="Filter-Precondition_confirmation-required"/>
        <separator/>
        <single-element-ref name="abort-condition" standalone="false"/>
        <macro-ref name="Abort-Condition_with_Comparison"/>
        <separator/>
        <single-element-ref name="complete-condition" standalone="false"/>
        <group-link name="Temporal Patterns"/>
      </group>
      <group name="Intentions">
        <macro-ref name="Intentions"/>
        <macro-ref name="Intention"/>
        <separator/>
        <group-link name="Temporal Patterns"/>
      </group>
      <group name="Plan-body">
        <single-element-ref name="plan-body"/>
        <separator/>
        <macro-ref name="Plan-Body_user-performed"/>
        <macro-ref name="Plan-Body_plan-activation"/>
        <macro-ref name="Plan-Body_Cyclical-Plan"/>
        <separator/>
        <macro-ref name="Plan-Body_subplans_plan-activation"/>
        <group name="Subplans">
          <single-element-ref name="subplans"/>
          <macro-ref name="Subplans"/>
          <macro-ref name="Subplans_ask"/>
          <macro-ref name="Subplans_plan-activation"/>
          <group name="Wait-For">
            [...]
          </group>
          <separator/>
          <group-link name="Plan-Body stuff"/>
        </group>
        <separator/>
        <group name="Plan-Body stuff">
          <single-element-ref name="user-performed"/>
          <macro-ref name="Plan-Activation"/>
          <separator/>
          <macro-ref name="Ask"/>
          <macro-ref name="Cyclical-Plan"/>
          <macro-ref name="Variable-Assignment_constant"/>
          <group-link name="Expressions"/>
          <separator/>
          <macro-ref name="If-then"/>
          <macro-ref name="If-then-else"/>
          <group-link name="Simple Conditions"/>
          <separator/>
          <macro-ref name="Refer-To"/>
          <single-element-ref name="to-be-defined"/>
        </group>
      </group>
      <separator/>
      <group name="Plan Stuff">
        <group name="Temporal Patterns">
          [...]
        </group>
        <macro-ref name="Plan-Group"/>
        <single-element-ref name="value-defs"/>
        <macro-ref name="Derived-From"/>
        <macro-ref name="Refers-To"/>
        <single-element-ref name="effects"/>
        <single-element-ref name="defaults"/>
        <single-element-ref name="arguments"/>
        <single-element-ref name="preferences"/>
        <single-element-ref name="returns" standalone="false"/>
        <separator/>
        <macro-ref name="Explanation"/>
        <group-link name="Expressions"/>
      </group>
    </group>
    <group name="Domains">
      <macro-ref name="Domain-Definition"/>
      <single-element-ref name="domain-defs"/>
      <separator/>
      <macro-ref name="Domain"/>
      <macro-ref name="Use-Domain"/>
      <separator/>
      <group name="Parameter Definitions">
        <macro-ref name="Parameter-Group"/>
        <macro-ref name="Parameter-Def"/>
        <group-link name="Parameter-Definition"/>
        <separator/>
        <group name="Raw-Data Definitions">
          [...]
        </group>
        <group name="Qualitative Parameter Definition">
          [...]
        </group>
        <macro-ref name="Change-Def"/>
        <single-element-ref name="parameter-ref"/>
      </group>
      <group-link name="Definitions"/>
    </group>
    <group name="Definitions">
      <group name="Record Definitions">
        <macro-ref name="Patient-Record-Def"/>
        <macro-ref name="Record-Def"/>
        <macro-ref name="Field-Def"/>
      </group>
      <macro-ref name="Qualitative-Scale-Def"/>
      <macro-ref name="Qualitative-Entry"/>
      <macro-ref name="Numerical-Scale-Def"/>
      <macro-ref name="Unit-Entry"/>
      <macro-ref name="Unit-Def"/>
      <separator/>
      <macro-ref name="Variable-Def"/>
      <macro-ref name="Constant-Def"/>
      <macro-ref name="Constant-Def_with-numerical-scalar-def"/>
      <macro-ref name="Iterator-Def"/>
      <macro-ref name="Iterator-Def_on-variable"/>
      <separator/>
      <group name="predefined Definitions">
        <macro-ref name="Unit-Def_mgdl"/>
      </group>
    </group>
    <group name="Patterns">
      <group name="Tables">
        <group name="Parameter-Definition">
          <macro-ref name="Param-Def-Table"/>
          <macro-ref name="Param-Def-Row"/>
          <macro-ref name="Param-Def-Column"/>
          <separator/>
          <group-link name="Domains"/>
        </group>
      </group>
    </group>
    <separator/>
    <macro-ref name="Library-Info"/>
    <single-element-ref name="library-defs"/>
    <group name="common Stuff">
      <group name="Expressions">
        [...]
      </group>
      <group name="any-comment">
        <macro-ref name="Link"/>
        <separator/>
        <macro-ref name="Comment"/>
        <macro-ref name="URL"/>
        <macro-ref name="BibRef"/>
        <single-element-ref name="not-yet-defined"/>
      </group>
      <single-element-ref name="administrative-data"/>
      <macro-ref name="Changes_Last-update"/>
      <macro-ref name="Changes"/>
    </group>
  </group>
</gmt>
```

Figure D.1: Sourcecode of `Asbru-Macros.gmt`

Bibliography

[Baumgartner et al., 2001]: Robert Baumgartner, Sergio Flesca and Georg Gottlob. *"Visual Web Information Extraction with Lixto"*. In: *Proceedings of the 27th Very Large Data Bases (VLDB) Conference*, Roma, Italy. 2001.
http://www.dbai.tuwien.ac.at/staff/baumgart/lixto/vldb.pdf

[Bechhofer, 2002]: Sean Bechhofer (editor). *"Ontology Language Standardisation Efforts"*. Deliverable 4.0 of the OntoWeb project, Information Management Group, Department of Computer Science, University of Manchester. 2002.
http://ontoweb.aifb.uni-karlsruhe.de/About/Deliverables/d4.0.pdf

[Bury et al., 2000]: Jonathan Bury, John Fox and David Sutton. *"The PROforma Guideline Specification Language: Progress and Prospects"*. In: Barbara Heller, Markus Löffler, Mark Musen and Mario Stefanelli (editors), *Computer-Based Support for Clinical Guidelines and Protocols - Proceedings of the First European Workshop on Computer-Based Support for Clinical Guidelines and Protocols (EGWLP) 2000*, Volume 83 of *Studies in Health Technology and Informatics*, pp. 12–29. IOS Press, 2001.
http://www.acl.icnet.uk/lab/PUBLICATIONS/ms364.pdf

[Corcho & Gómez, 2000]: Oscar Corcho and Asunción Gómez-Pérez. *"A Roadmap to Ontology Specification Languages"*. Facultad de Informática, Universidad Politécnica de Madrid. Spain. 2000.
http://delicias.dia.fi.upm.es/articulos/ocorcho/ekaw2000-corcho.pdf

[Friedland & Iwasaki, 1985]: P. Friedland and Y. Iwasaki. *"The Concept and Implementation of Skeletal Plans"*. In: *Journal of Automated Reasoning* 1(2) (1985) 161–208.

[Geldof, 2002]: Marije Geldof. *"The formalization of medical protocols: easier said than done"*. Master's Thesis, Department of Artificial Intelligence, Vrije Universiteit, Amsterdam. December 2002.

[Gruber, 1992]: Thomas R. Gruber. *"A Translation Approach to Portable Ontology Specifications"*. Technical Report KSL 92-71, Knowledge Systems Laboratory, Computer Science Department, Stanford University. 1992.
ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/KSL-92-71.ps

[Jonassen et al., 1993]: David H. Jonassen, Katherine Beissner and Michael Yacci. *"STRUCTURAL KNOWLEDGE – Techniques for Representing, Conveying, and Acquiring Structural Knowledge"*. Published by Lawrence Erlbaum Associates, Hillsdale. 1993.

[Kosara, 1999]: Robert Kosara. *"Metaphors of Movement — A User Interface for Manipulating Time-Oriented, Skeletal Plans"*. Master's Thesis, Institute of Software Technology and Interactive Systems, Vienna University of Technology, Austria. 1999.
http://www.kosara.net/papers/KosaraMastersThesis.pdf

[Kosara & Miksch, 2001]: Robert Kosara and Silvia Miksch. *"Metaphors of Movement: A Visualization and User Interface for Time-Oriented, Skeletal Plans"*. In: *Artificial Intelligence in Medicine*, Special Issue: Information Visualization in Medicine, pp. 111-131, 22(2), 2001.
http://www.ifs.tuwien.ac.at/~silvia/pub/publications/kos_aimj2000.pdf

[Kosara et al., 2000]: Robert Kosara, Klaus Hammermüller and Silvia Miksch. *"Co-Designing XML-based Languages and Classes with* Pontifex". Institute of Software Technology and Interactive Systems, Vienna University of Technology, Austria. Technical Report, Asgaard-TR-2000-2. 2000.
http://www.ifs.tuwien.ac.at/asgaard/Technical-Reports/Asgaard-TR-2000-1.pdf

[Kosara et al., 2002]: Robert Kosara, Silvia Miksch, Andreas Seyfang and Peter Votruba. *"Tools for Acquiring Clinical Guidelines in Asbru"*. In: *Proceedings of Sixth World Conference on Integrated Design and Process Technology (IDPT'02)*. 2002.
http://www.kosara.net/papers/IDPT2002Kosara.pdf

[Messner, 2000]: Peter Messner. *"Time Shapes - A Visualization for Temporal Uncertainty in Planning"*. Master's Thesis, Institute of Software Technology and Interactive Systems, Vienna University of Technology, Vienna, Austria. 2000.
http://www.ifs.tuwien.ac.at/asgaard/asbru/tools/asbruview/timeshapes.pdf

[Miksch et al., 1997a]: Silvia Miksch, Yuval Shahar and Peter Johnson. *"Asbru: A Task-Specific, Intention-Based, and Time-Oriented Language for Representing Skeletal Plans"*. In: E. Motta, F. v. Harmelen, C. Pierret-Golbreich, I. Filby, N. Wijngaards (eds.), *Proceedings of the 7th Workshop on Knowledge Engineering: Methods & Languages (KEML-97)*. Milton Keynes, UK, Open University. 1997.
http://www.ifs.tuwien.ac.at/~silvia/pub/publications/mik_keml97.pdf

[Miksch et al., 1997b]: Silvia Miksch, Yuval Shahar, Werner Horn, Christian Popow, Franz Paky and Peter Johnson. *"Time-Oriented Skeletal Plans: Support to Design and Execution"*. In: S. Steel, R. Alami (eds.), *Recent Advances in AI Planning - Fourth European Conference on Planning ECP '97 (EWSP'97)*, Springer, Lecture Notes in Artificial Intelligence (LNAI), Toulouse, France. 1997.
http://www.ifs.tuwien.ac.at/~silvia/pub/publications/mik_ecp97.pdf

[Miksch et al., 1998]: Silvia Miksch, Robert Kosara, Yuval Shahar and Peter Johnson. *"AsbruView: Visualization of Time-Oriented, Skeletal Plans"*. In: R. Simmons, M. Veloso, S. Smith (eds.), *Proceedings of the fourth International Conference on Artificial Intelligence Planning Systems 1998 (AIPS-98)*. Carnegie Mellon University, AAAI Press. 1998.
http://www.ifs.tuwien.ac.at/~silvia/pub/publications/mik_aips98.pdf

[Musen et al., 1996]: Mark A. Musen, Samson W. Tu, Amar K. Das and Yuval Shahar. *"EON: A Component-Based Approach to Automation of Protocol-Directed Therapy"*. Section on Medical Informatics, Stanford University School of Medicine. In: *Journal of the American Medical Informatics Society (JAMIA)*, 3: pp. 367-388. 1996.
http://smi-web.stanford.edu/pubs/SMI_Reports/SMI-96-0606.pdf

[Ohno-Machado et al. 1998]: Lucila Ohno-Machado, John H. Gennari, Shawn Murphy, Nilesh L. Jain, Samson W. Tu, Diane E. Oliver, Edward Pattison-Gordon, Robert A. Greenes, Edward H. Shortliffe and G. Octo Barnett. *"The GuideLine Interchange Format: A Model for Representing Guidelines"*. In: *Journal of the American Medical Informatics Society (JAMIA)*, 5(4): pp. 357-372. 1998.
http://www-smi.stanford.edu/pubs/SMI_Reports/SMI-97-0668.pdf

[Peleg et al., 2002a]: M. Peleg, S. Tu, J. Bury, P. Ciccarese, J. Fox, R. A. Greenes, R. Hall, P. D. Johnson, N. Jones, A. Kumar, S. Miksch, S. Quaglini, A. Seyfang, E. H. Shortliffe and M. Stefanelli. *"Comparing Models of Decision and Action for Guideline-Based Decision Support: a Case-Study Approach (Part 1 of 2)"*. Stanford Medical Informatics, Technical Report SMI-2002-0922. 2002.
http://www.smi.stanford.edu/pubs/SMI_Reports/SMI-2002-0922.pdf

[Peleg et al., 2002b]: M. Peleg, S. Tu, J. Bury, P. Ciccarese, J. Fox, R. A. Greenes, R. Hall, P. D. Johnson, N. Jones, A. Kumar, S. Miksch, S. Quaglini, A. Seyfang, E. H. Shortliffe and M. Stefanelli. *"Comparing Models of Decision and Action for Guideline-Based Decision Support: a Case-Study Approach (Part 2 of 2)"*. Stanford Medical Informatics, Technical Report SMI-2002-0923. 2002.
http://www.smi.stanford.edu/pubs/SMI_Reports/SMI-2002-0923.pdf

[Plaisant et al., 1996]: Catherine Plaisant, Brett Milash, Anne Rose, Seth Widoff and Ben Shneiderman. *"LifeLines: Visualizing Personal Histories"*. In: *Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems*, volume 1 of *PAPERS: Interactive Information Retrieval*, pages 221–227. 1996.
http://www.cs.umd.edu/Library/TRs/CS-TR-3523/CS-TR-3523.ps.Z

[Polvani et al., 2000]: Kristi-Anne Polvani, Abha Agrawal, Bryant Karras, Aniruddha Deshpande and Richard Shiffman. *"GEM Cutter Manual"*. Yale Center for Medical Informatics. 2000.
http://ycmi.med.yale.edu/GEM/GEM_%20cutter/GEM%20Cutter%20Manual.pdf

[Purves et al., 1999]: Ian N. Purves, Bob Sugden and Nick Booth. *"The PRODIGY project - the iterative development of the release one model"*. In: *Proceedings of AMIA Symposium 1999*; pp. 359-63. 1999. http://www.amia.org/pubs/symposia/D005532.PDF

[Roomans, 2001]: Hugo F. Roomans *"Formalisation of a medical guideline - Usability investigation of the ASBRU modeling language"*. Master's Thesis, Vrije Universiteit, Amsterdam. 2001. http://www.protocure.org/Projects/HugoThesis.pdf

[Schreiber et al., 1994]: Guus Schreiber, Bob Wielinga, Robert de Hoog, Hans Akkermans and Walter van de Velde. *"CommonKADS: A Comprehensive Methodology for KBS Development"*. In: IEEE Expert, 1994.
http://www.cs.dal.ca/~riordan/CSCI6501/CommonKads.pdf

[Seyfang et al., 2002]: Andreas Seyfang, Robert Kosara and Silvia Miksch. *"Asbru's Reference Manual, Asbru version 7.3"*. Institute of Software Technology, Vienna University of Technology, Technical Report, Asgaard-TR-2002-1. 2002.
http://www.ifs.tuwien.ac.at/asgaard/asbru/asbru_7_3/asbru_7.3_reference.pdf

[Shahar et al., 1998]: Yuval Shahar, Silvia Miksch and Peter Johnson. *"The Asgaard Project: A Task-Specific Framework for the Application and Critiquing of Time-Oriented Clinical Guidelines"*. In: *Artificial Intelligence in Medicine*, 14, pp. 29-51. 1998. http://www.ifs.tuwien.ac.at/~silvia/pub/publications/sha_aimj98.pdf

[Shiffman et al., 2000]: Richard N. Shiffman, Bryant T. Karras, Abha Agrawal, Roland Chen, Luis Marenco and Sujai Nath. *"GEM: A proposal for a more comprehensive guideline document model using XML"*. In: *Journal of the American Medical Informatics Society (JAMIA)*, 7: pages 488–498. October 2000. http://ycmi.med.yale.edu/GEM/Publications/GEM-jamia-sept200-fulltext.pdf

[Steele & Fox, 2002]: Rory Steele and John Fox. *"Tallis PRO*forma *Primer - Introduction to PRO*forma *language and software with worked examples"*. Advanced Computation Laboratory, Cancer Research, UK. May 2002. http://www.acl.icnet.uk/lab/tallisPrimer.pdf

[Svatek et al., 2000]: Vojtěch Svátek, Tomáš Kroupa, and Marek Růžička. *"Guide-x – a step-by-step, markup-based approach to guideline formalisation"*. In: Barbara Heller, Markus Löffler, Mark Musen, and Mario Stefanelli (editors), *Computer-Based Support for Clinical Guidelines and Protocols - Proceedings of the First European Workshop on Computer-Based Support for Clinical Guidelines and Protocols (EGWLP) 2000*, Volume 83 of *Studies in Health Technology and Informatics*, pages 97–114. IOS Press. 2001. http://nb.vse.cz/~svatek/ewglp00.ps

[Toprakkiran, 2001]: Sibel Toprakkiran. *"Ontologies in Lixto - Research and Implementation Issues"*. Master's Thesis, Database and Artificial Intelligence Group, Institute of Information Systems, Vienna University of Technology, Austria. 2001. http://www.dbai.tuwien.ac.at/proj/lixto/concepts.zip

[Vollebregt et al., 1999]: Arjen Vollebregt, Annette ten Teije, Frank van Harmelen, Johan van der Lei and Mees Mosseveld. *"A study of PROforma, a development methodology for clinical procedures"*. In: *Artificial Intelligence in Medicine*, 17, pp. 195-221. 1999. http://www.cs.vu.nl/~frankh/postscript/AIM99.pdf

[Votruba, 2003]: Peter Votruba. *"Guideline Markup Tool – User Manual"*. Institute of Software Technology, Vienna University of Technology, Technical Report - Asgaard-TR-2003-1. 2003.

Related Websites

A List of related Websites grouped by the chapter to which they belong or where they were first mentioned:

- **Chapter 1 - *Introduction*** (p. 1)**:**
  [URL #1] Asgaard Project: http://www.asgaard.tuwien.ac.at/

- **Chapter 2 - *Definitions of basic Terms*** (p. 3)**:**
  [URL #2] OpenClinical – Glossary: http://www.openclinical.org/glossary.html
  [URL #3] SCHEMAS – Glossary: http://www.schemas-forum.org/related/glossary.html
  [URL #4] OpenClinical - Clinical practice guidelines: http://www.openclinical.org/guidelines.html
  [URL #5] AAP (American Academy of Pediatrics) - Practice Guidelines: http://www.aap.org/policy/paramtoc.html

- **Chapter 3 - *Structured Knowledge Acquisition and Modelling*** (p. 4)**:**
  [URL #6] CommonKADS: http://www.commonkads.uva.nl/
  [URL #7] World Wide Web Consortium (W3C): http://www.w3.org/
  [URL #8] DAML+OIL (March 2001): http://www.daml.org/2001/03/daml+oil-index.html
  [URL #9] W3C's DAML+OIL (March 2001) Reference Description: http://www.w3.org/TR/daml+oil-reference
  [URL #10] Annotated DAML+OIL (March 2001) Ontology Markup: http://www.daml.org/2001/03/daml+oil-walkthru
  [URL #11] DARPA Agent Markup Language (DAML): http://www.daml.org/
  [URL #12] Ontology Inference Layer (OIL): http://www.ontoknowledge.org/oil/index.shtml
  [URL #13] W3C's Resource Description Framework (RDF): http://www.w3.org/RDF/
  [URL #14] Lixto at TU-Vienna: http://www.dbai.tuwien.ac.at/proj/lixto/
  [URL #15] Lixto Software GmbH: http://www.lixto.com/

- **Chapter 4 - *Guideline Modelling Methods*** (p. 8)**:**
  [URL #16] Asbru: http://www.asgaard.tuwien.ac.at/asbrusyntax.html
  [URL #17] Protocure - Development of Asbru guidelines: http://www.protocure.org/
  [URL #18] EON: http://smi-web.stanford.edu/projects/eon/
  [URL #19] OpenClinical – EON: http://www.openclinical.org/gmm_eon.html
  [URL #20] GEM: http://ycmi.med.yale.edu/GEM/
  [URL #21] OpenClinical – GEM: http://www.openclinical.org/gmm_gem.html
  [URL #22] GLARE: http://www.openclinical.org/gmm_glare.html
  [URL #23] GLIF.ORG: http://www.glif.org/
  [URL #24] OpenClinical – GLIF: http://www.openclinical.org/gmm_glif.html
  [URL #25] OpenClinical – GUIDE: http://www.openclinical.org/gmm_guide.html
  [URL #26] Prodigy: http://www.prodigy.nhs.uk/
  [URL #27] OpenClinical – Prodigy: http://www.openclinical.org/gmm_prodigy.html
  [URL #28] PRO*forma*: http://www.acl.icnet.uk/lab/proforma.html

[URL #29] OpenClinical – PRO*forma*:
http://www.openclinical.org/gmm_proforma.html

- **Chapter 5 - *Guideline Acquisition*** (p. 12)**:**
[URL #30] AsbruView: http://www.asgaard.tuwien.ac.at/asbruview/
[URL #31] W3C - XML Path Language (XPath): http://www.w3.org/TR/xpath
[URL #32] GEM Cutter:
http://ycmi.med.yale.edu/GEM/GEM_%20cutter/gem_cutter.htm
[URL #33] PRO*forma* – Arrezzo:
http://www.acl.icnet.uk/lab/proformatechnologies.html
[URL #34] PRO*forma* – TALLIS: http://www.acl.icnet.uk/lab/tallis.html
[URL #35] Protégé Project: http://protege.stanford.edu/

- **Chapter 6 - *Summary of presented projects*** (p. 20)**:**
[URL #36] OpenClinical - Comparison of guideline modelling methods:
http://www.openclinical.org/gmmcomparison.html

- **Chapter 7 - *The Guideline Markup Tool*** (p. 21)**:**
[URL #37] GMT: http://www.asgaard.tuwien.ac.at/~peter/GMT/
[URL #38] Apache's Xerces: http://xml.apache.org/xerces2-j/
[URL #39] Apache's Crimson: http://xml.apache.org/crimson/
[URL #40] Apache's Xalan: http://xml.apache.org/xalan-j/
[URL #41] Saxon: http://saxon.sourceforge.net/

- **Chapter 8 - *The Macros File*** (p. 34)**:**
[URL #42] GMT - Macros page:
http://asgaard.tuwien.ac.at/~peter/GMT/macros/

- **Chapter 11 - *Future Work*** (p. 44)**:**
[URL #43] GMT – Status page / List of To-Do's:
http://www.asgaard.tuwien.ac.at/~peter/GMT/status.html

- **Appendix B - *Installation Instructions of the GMT*** (p. 51)**:**
[URL #44] GMT - Download page:
http://www.asgaard.tuwien.ac.at/~peter/GMT/download/
[URL #45] Java Downloads: http://java.sun.com/j2se/1.4.1/download.html
[URL #46] Java Web Start (JWS) – Download page:
http://java.sun.com/cgi-bin/javawebstart-platform.sh?