



IST-FP6-508794

PROTOCURE II

*Integrating formal methods in the development process of
medical guidelines and protocols*

Specific Targeted Research Project
Information Society Technologies

D2.2a Specification of formats of intermediate, Asbru to KIV representations.

Due date of deliverable: 31 December 2004
Actual submission date: 21 December 2004
Submission date of revision: 5 September 2005

Start date of project: 1 January 2004 **Duration:** 30 months

Organisation name of lead contractor: Universitat Jaume I

Revision 2

Project co-funded by the European Commissions within the Sixth Framework Programme(2002-2006)		
Dissemination Level		
PU	Public	√
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

D2.2a Specification of Formats of Intermediate, Asbru and KIV Representations

Document Version 1.2

Andreas Seyfang, Silvia Miksch, Peter Votruba
Vienna University of Technology, Austria

Kitty Rosenbrand, Jolanda Wittenberg, Joyce von Croonenborg
Dutch Institute for Healthcare Improvement, the Netherlands

Wolfgang Reif, Michael Balser, Jonathan Schmitt
University of Augsburg, Germany

Theo van der Weide, Peter Lucas, Arjen Homersom
University of Nijmegen, the Netherlands

September 2, 2005

Abstract

This document specifies the representations used to represent clinical guidelines in the Protocure II project. The original guideline is in English. It is first translated to the intermediate representation MHB. This version is translated to the plan representation Asbru which is again translated to the temporal logic KIV.

Asbru and KIV have already been applied in Protocure I. To complement them, an intermediate representation was designed which bridges the gap between the original guideline in English and Asbru. It models the content of the guideline as groups of chunks. Each chunk has different aspects which are structured in eight dimensions. The development of this new representation is described in detail in this document. For Asbru and KIV we give an overview and refer to the existing corpus of literature.

Contents

1	Specification of Intermediate Representation	3
1.1	Introduction	3
1.1.1	Motivation	3
1.1.2	Aims	4
1.2	Background	4
1.2.1	Development background	4
1.2.2	Structure and content of natural language guidelines	7
1.2.3	The AGREE instrument	10
1.2.4	Other Formal Representations of Guidelines	12
1.3	Details on the Intermediate Representation – The Many-Headed Bridge	16
1.3.1	Control flow	20
1.3.2	Data flow	25
1.3.3	Temporal aspects	26
1.3.4	Evidence	30
1.3.5	Background information	32
1.3.6	Resources	33
1.3.7	Patient related aspects	35
1.3.8	Document structure	35
1.4	Evaluation	37
2	Specification of Asbru Version	38
2.1	Plan Library	38
2.1.1	Global Declarations	39
2.2	Domain Definition	39
2.2.1	Type Definitions	39
2.2.2	Parameter Definition	39
2.2.3	Other Parts of Domain Definition	41
2.3	Plans	41
2.4	Plan Body	41
2.5	Basic Elements	42
2.5.1	Simple Condition	42
2.5.2	Time Annotation	42
2.5.3	Temporal Pattern	43
2.5.4	Expression	43
3	Specification of Asbru in KIV	44
3.1	Introduction	44
3.2	XML Input Format for KIV	44
3.3	Algebraic Specifications	49
3.3.1	Example	49
3.3.2	Syntax specification	51
3.4	XSL Transformation	52

3.5	Open Issues	52
3.5.1	Changes to previous version of this paper	53
3.6	DTD for Asbru plans in KIV	53

Chapter 1

Specification of Intermediate Representation

1.1 Introduction

This chapter describes the background and details of the intermediate representation which was developed as part of workpackage 2 of the Protocure II project.

The remainder of this section gives motivation and aims for this development. Section 1.2 describes the background. Section 1.3 describes the intermediate representation in full detail. Finally, Section 1.4 describes performed and planned evaluation of this representation.

1.1.1 Motivation

To bridge the gap between original and Asbru version. Previous experience in translating a guideline into Asbru (e.g., during Protocure I) showed that it is very difficult to accomplish this task in a single step. Therefore, the first aim is to define (at least) one intermediate step in the transformation of a guideline written in natural language (narrative, tables, figures) into Asbru with keeping further translation to KIV in mind. A natural solution is the invention of an *intermediate representation* which represents the content more concise and structured than natural language but less precise than Asbru or another formal guideline representation.

To be independent from the formal representation used. Related to the first aim is the assumption that the intermediate representation would be independent from the formal guideline representation, which is Asbru in this project. This should decrease the amount of training required for the person who creates the intermediate model. It also would allow for a separation of the issue of creating a (more) formal model of the guideline from Asbru-specific issues. This hope implies that some difficulties are induced by Asbru and which are not present in other representations or which need not be addressed in a less precise representation.

An intermediate representation covering the capabilities of several formal representations would form a suitable basis to discuss the strength and limitations of different guideline representations for a given guideline. However, such a comparison is not part of Protocure II. Instead, we show how the intermediate representation described in this document maps to different guideline representations.

To support human-readable, living guidelines. In the Protocure II project we develop support for the development of living guidelines. Therefore, the target of the modeling process is both a formal representation of the guideline and a printed guideline in natural language in a

format optimal for the distribution among human readers¹. At the same time, we believe that formal modeling helps to improve the quality of a guideline. As a consequence, the modeling process becomes bi-directional – from informal to formal and back to informal.

Another important aspect in related to living guidelines is computer support for frequent revisions and different versions of the guideline to minimize the work load associated with building a new version of the guideline based on an existing one.

1.1.2 Aims

The threefold motivation above results in a broad range of aims. They are described in the following in the order of the corresponding motivation. Note that many of these aims are contradicting – therefore any solution will be a compromise between them.

Simplicity. Annotating a guideline should be easy to learn and the resulting representation should be easy to understand by persons not familiar with the syntax.

Richness. While being simple to understand, the representation must cover all relevant aspects of a guideline in a degree of detail which significantly supports the creation of a formal version (e.g., in Asbru) afterwards.

Support of dual structure of guideline. The structure of the natural language text version of a guideline is optimized for human readers. The structure of the formal version of a guideline is organized according to the procedural decomposition of the described processes. These structures do not meet at all. The structure of the intermediate representation must, therefore, close this gap in a flexible way.

1.2 Background

This section describes the background for the development of the intermediate representation.

Section 1.2.1 contains various issues of the Protocure II project which are relevant for the intermediate representation. Section 1.2.2 explores the structure and content of the type of guidelines modeled in this project. Section 1.2.3 describes the AGREE instrument for the appraisal of guideline quality. It forms an important background for the work described here, since one aim is to improve the quality of guidelines. Section 1.2.4 describes existing formal representations for guidelines other than Asbru. Our intermediate representation is designed to bridge natural language guidelines to these, too.

1.2.1 Development background

This section briefly describes various aspects which influence the work described here. First, the plan representation language Asbru is used in Protocure II. It is described in detail in Chapter 2 of this deliverable. Still, the most important aspects are given in the following subsection.

Second, to understand the work with the intermediate representation, the tool which is used – the Guideline Markup Tool (GMT) – is briefly described. See deliverable D2.3a for a full description of the GMT.

Third, some aspects of the guideline design process which are relevant for the design of the intermediate representation are described. The development process of living guidelines is explored in detail in deliverable D1.3.

¹For the presented work it is not important whether the reader of the guideline is a physician, a nurse, or a patient. It is of course for the authors of the guideline who will be well aware of the different background of different groups of readers. However, this does not influence issues of intermediate representation. Support for the authoring of parallel guideline versions for different groups of readers is not within the scope of our project. Also not in the scope of this project is *natural language processing* or *natural language generation*.

Asbru

The formal representation used in Protocure I was Asbru light, i.e., a subset defined for the needs of that project.

Asbru is designed to represent protocols rather than guidelines. Guidelines are "Systematically developed statements to assist practitioner and patient decisions about appropriate healthcare for specific clinical circumstances" [4]. Protocols are local tools that set out specifically what should happen, when and by whom in the care process. They can be seen as the local definition of a particular care process derived from a more discretionary guideline. Protocols reflect local circumstances, and variation will be due to the differing types of local provision. Although there is a difference between protocols and guidelines, Protocure I proved that not only protocols but also guidelines can be represented Asbru.

Chapter 2 contains an introduction to the parts of Asbru. Full details can be found in [9].

The Guideline Markup Tool – GMT

Authoring and maintenance of guidelines need methods and tools to support them. The tool which is used to generate and maintain the guideline's version in intermediate representation forms an important background for the design of the representation itself. This tool is called Guideline Markup Tool (GMT) developed during Protocure II.

GMT is a tool that helps translating guidelines in free text into Asbru, by providing two main features: (i) linking between a textual guideline and its formal representations, and (ii) applying design patterns in the form of macros. Firstly, GMT allows the definition of links between the original guideline and the intermediate representation, which gives the user the possibility to find out where a certain value in the Asbru notation comes from. Therefore, if someone wants to know the origin of a specific value in the XML file containing the intermediate representation, the GMT can be used to jump to the correlating point in the HTML file where the value is defined and the other way round.

The second main feature of the GMT is the usage of macros. A macro combines several XML elements, which are usually used together. Thus, using macros allows creating and extending XML containing intermediate representation files more easily through the usage of common design patterns. Such design patterns are often used behaviors, which can be found in guidelines.

Through these two features, GMT is able to support the following tasks:

Authoring and augmenting guidelines. We want to be able to take a new guideline in plain text and create an (XML-based) intermediate representation of it, and to add links to the corresponding parts of a guideline to an already existing XML file.

Understanding intermediate representations of guidelines. For an guideline in intermediate representation, we want to be able to see where values in the different parts of the intermediate representation come from, and how parts of the original text were translated into it. This is important not just for knowledge engineers, but also for physicians wanting to develop an understanding of the intermediate representation.

Structuring the syntax. The GMT provides a structured list of elements of the target language, which is the intermediate representation in our case – the macros – that needs to be done in a way that best supports the authoring of plans. This list will also provide a good starting point for teaching material and possible subsets of the language for special purposes.

These feature releases the intermediate representation from two objectives: First, the original text parts need not be stored as part of the intermediate representation elements. Instead, the links clearly show the source of each element in the intermediate representation. Second, there is no need to produce a guideline in natural language from the intermediate representation, since the original text remains unaltered, except for the insertion of links which are easily hidden.

Details on the GMT are given in Deliverable D2.3a "Mark-up and editing tool" available at www.protocure.org.

Process-related aspects

The transformation of a narrative guideline into a formal representation such as Asbru is performed in three steps.

1. A guideline developer produces intermediate representation from the original text guideline, marking up every bit of the guideline in as much detail as he or she can find.
2. A knowledge engineer together with a guideline developer who is familiar with clinical algorithms complements the intermediate representation to form a complete algorithm. At this time, missing information is acquired from physicians and added in an appendix to the original guideline, therefore clearly maintaining the difference in status of the two sources of information.
3. A knowledge engineer familiar with the target formal representation (e.g., Asbru) builds a formal model of the guideline based on the intermediate representation.

In practice, several back loops will be necessary, but their amount will clearly decrease with increased familiarity with the process and the demand from the further steps in the process. E.g., it is unlikely that a person without any knowledge of Asbru will provide all the required information in step 2, but it is very likely that the effort of learning about the requirements for Asbru modeling is a fraction of learning Asbru.

The guideline is represented in four different forms: natural language, intermediate representation, Asbru, and KIV. For each of the three pairs of neighboring representations in this line of abstraction, parts of each side can be associated with each other using the Guideline Markup Tool. It inserts a special markup into each of the files which contains a reference to the other side.

The natural language version of the guideline is structured in a way which accommodates the readers, i.e., developers and users of the guideline. In contrast, the Asbru version of a guideline is structured as a hierarchy of plans with a top-level plan representing the activity described in the guideline and sub-plans describing the steps to perform in increasing levels of detail. Therefore, the structure of this version of the guideline is dictated by the logical layout of the processes taking place.

The layout of the guideline in intermediate representation can be freely chosen by the knowledge engineer. It will first follow the original text of the guideline closely. After a first walk-through, the concepts and processes described will be refined in order to make them precise enough for formal representation and to close gaps in the original guideline. In the course of this process, parts of the guideline in intermediate representation will be moved around to follow the logical structure of the guideline. Still the connection to the natural language text will always be clear due to the special links between the two documents. The natural language text version is not altered, except for the insertion of links (which can be hidden from a reader).

If the analysis during the modeling process hints at some rearrangement of the original text, it can immediately take place without any consequences on the links².

Guidelines often lack information which is necessary to form a complete formal model of the described topic, as would be necessary to build a domain model. This is not an error, since such information is assumed to be known by any person reading and applying the guideline. Still, it is not available to the computer program executing or verifying the guideline. And it is not the aim of the guideline formalization process to create such a domain model.

Most naturally, such added knowledge must be marked as not being part of the guideline. And it should be described as precise and comprehensive as possible. Both can be achieved by adding natural language text containing additional information as an appendix to the original guideline. This shows the domain experts reviewing the guideline, which assumptions were made in building

²If a group of words is linked to a concept in intermediate representation and only some of these words are moved to another place, then both parts of the word group will be linked to the concept.

the formal model, i.e. these assumptions undergo reviewing, too (if somewhat more informal). In the intermediate representation version of the guideline, links to the appendix are formally treated just as others on the technical level, but pointing to the appendix distinguishes them from those links pointing to the guideline proper.

As a result of the discussion process, some parts of the appendix might become part of the official guideline, which – on the technical level – would be just a move of paragraphs in the natural language document.

The same holds for clinical algorithms (also called flowcharts) which may first be represented in intermediate representation in order to match the demands of the more formal Asbru representation. Such parts of intermediate representation should also be linked to comments in the unofficial appendix. If the algorithm is included in the official guideline latter, the comments from the appendix will be restated if necessary and moved to an official part of the guideline. As of this writing it is not planned to support any XML representation of flowcharts but they can be described fair enough in HTML to guide the person drawing them for the final version of the guideline.

1.2.2 Structure and content of natural language guidelines

This section describes the structure and general features of the content of guidelines we are dealing with in our project. The guideline modeled during Protocure II is the Dutch Guideline for the Treatment of Breast Carcinoma [6]. Therefore, the following observations are mainly based on breast cancer guidelines from CBO, with backing from other guidelines we previously modeled or studied.

Structure-related aspects

Information in a guideline appears in various forms.

Narrative text. Concerning the quantity, the most important part of a guideline is narrative text. It contains the scientific justification and other considerations for the recommendations, historical and general comments. The reason behind this text is to assure the correctness and appropriateness of the recommendations to the interested reader. Narrative text will be found mainly in the paragraphs named: "scientific justification" and "other considerations."

Summary statements of the evidence. The heart of an evidence-based guideline are concise statements for which there is a certain, clearly defined evidence. They consist of a short paragraph, an indication of the grade of evidence for the statement in the paragraph, and a list of references to literature, grouped by the degree of evidence they provide. The styling of each summary statement is designed in a way which makes it stand out of the surrounding text and which separates one statement from the other.

Recommendations. This is the most important part of the guideline. Summarizing the narrative text, the recommendations form less than one page per chapter of the guideline. They are free text, with a relatively high fraction of bullet lists. The styling of each recommendation is designed in a way which makes it stand out of the surrounding text.

Diagrams of clinical algorithms. Some guidelines contain diagrams – often referred to as flowcharts – which display important or easy to algorithmically structure parts of the clinical algorithm. While these diagrams rarely match the precision of flowcharts in computer science, the guideline authors successfully use them to point out important aspects in an algorithmical way.

Literature references. At the end of each chapter or at the end of the guideline, the used literature is listed in a standard way (e.g. Vancouver style), without additional information about content or degree of evidence.

Tables. Some guidelines contain tables to summarize important aspects (similar to a clinical algorithm) or to provide a large amount of detailed information in an efficient way, such as drug administration.

Evidence Tables. Some guidelines contain tables to summarize the scientific literature used as justification for the recommendations. This provides a large amount of detailed information in an efficient way.

Illustrations. In some cases there are various illustrations depicting something which is described in the text.

Analyzing the content

Being aimed at synchronizing the knowledge of their readers, the guideline typically spans a bow from the importance of the topic and recent developments to concrete discussion of diagnosis and therapy, finishing with organizational, psychological and other aspects.

Dimensions of statements. In each part, each statement plays one or more roles in the narrative or explanation process. Furthermore the contained information can be interpreted on various levels. The *dimensions* of each statement which are described in the following are a way to break down its complexity into manageable parts. While this section only gives a brief overview, Section 1.3 features a detailed discussion of each of them.

Control flow. One aspect of a guideline is that it tells the reader *when* to do *what*. Both when and what need not be too precise, but it is clear that these pieces of information together form a directive for the reader's actions. The completeness and vagueness of these directions forms an important part of our research, since often it is impossible or not advisable to specify action too precisely, while on the other hand unambiguous directives are an attractive goal. Examples are given in Section 1.3.

Data dimension. The description of the data processing is interwoven with control flow. It is involved both in the diagnosis and treatment of the patient. While in descriptions of the treatment of diseases the necessary information is often implicitly assumed to be available, the diagnosis part often contains detailed descriptions. While control flow describes the activities for gathering of information, data flow describes how one piece of information is abstracted from other ones.

Temporal dimension. Both data and control flow may have temporal aspects. These refer to the time during which an action should be taken, or for which a certain abstraction is valid.

Modeling the temporal dimension of processes, measurement, effects, and intentions is one of the outstanding features of Asbru and we are confident to improve the quality of protocols and guidelines by making this often implicit information explicit.

Evidence. Many, but not all statements in an evidence-based guideline contain references to literature which provides the evidence for them. These references may or may not be annotated by a certain grade of evidence. Another form of expressing the degree of evidence is the usage of words like "can" or phrases such as "is likely" (compare example 3 below).

Background information. Guidelines contain a considerable amount of information which is neither directly nor indirectly part of a directive of actions or analysis steps (in diagnosis). Instead, it informs the reader in a general way. The rational behind it ranges from motivation of the reader to follow the recommendations to didactical considerations.

Resources. Each action in diagnosis and treatment consumes resources: It takes time of health care staff, it causes patient discomfort of varying degree, and it bears financial cost. In this context,

”doing nothing”, i.e., waiting is considered as an action. The same holds for diagnosis steps. On the other side, the health of the patient can be considered as a resource which is improved by most treatment action – often at distinct levels.

Document structure. While the position of a sentence in the guideline document could be considered to be a matter of layout only, its status (scientific justification, evidence-based conclusion, recommendation) certainly forms an important context of its interpretation.

Diverse knowledge modeling aspects. There are several aspects in the statements of a guideline beyond those listed above which need to be pointed out.

Negative knowledge. Statements about the inappropriateness of certain actions form an important part of a guideline. Also, statements about lack of evidence are important information as in example 1 below.

Incomplete knowledge. Large parts of the medical knowledge are known in their qualitative form but not in quantity (compare example 2 below). It is important to know this fact and to distribute it in guidelines, but neither the importance of the contribution, nor the certainty or the extent of the effect are known.

Non-imperative conclusions or recommendations. Example 3 shows a conclusion which is not imperative. While it is easy to include a series of tokens for various forms of vague conclusions, such as ”can” and ”seen important”, their translation into any formal representation will cause problems. While this can be considered a problem of the formal representation, not of the intermediate representation, it is worthwhile remembering that nuances defined in the intermediate representation may get lost in the further translation to a formal representation.

Events and actions. Some proceedings have identifiable actors such treatment steps – they are called *actions* in this document. Others do not have personalized actors a disease, instead they are caused by cells or chemicals – they are called *events* in this document. Note that the temporal dimension of these events is not described by a time point but by intervals, i.e., events have durations – sometimes of significant extend.

The guideline most naturally describes both events and actions together. In a clearly written guideline, it will always be obvious to which event or action a certain detail such as an observed effect relates to. When modeling anything in the guideline as one group of plans (or happenings), it can be difficult to describe the often unknown relations of all events and actions to each other. While this again is mostly a problem of formal modeling, it will also appear when creating a clinical algorithm using the intermediate representation.

Example 1 *The five randomized clinical trials which investigated the value of adjuvant and/or neoadjuvant treatment compared with locoregional treatment alone for unresectable locoregionally advanced breast cancer, could find no survival benefit, not even in the long-term.* First sentence of Section 3.3 in [6]

Example 2 *Adjuvant hormone therapy in locally advanced breast cancer results in improved survival in the long-term.* Section 3.3 in [6], last conclusion

Example 3 *Physio-therapeutic intervention in the first month after axillary gland dissection can be effective.* Section 1.4, first evidence-based conclusion in [6]

1.2.3 The AGREE instrument

The AGREE instrument [3] was developed by a collaboration of guideline developing organizations from ten European countries. It provides a framework for assessing the quality of clinical practice guidelines and represents the current state of art of guideline evaluation in Europe.

One way to look at the intermediate representation is to see it as a means to improve guideline quality and therefore increase the score of a guideline in quality assessments. The following examines the demands on a guideline set forth by the AGREE instrument.

The guideline is rated as a whole for 23 items which are grouped in six domains. Some of these items are related to organizational issues such as stakeholder involvement which are beyond the scope of our project. Others are more or less strongly related to representation issues.

In the following, each of the domains is briefly introduced and for those items, which relate to our work, the potential influence on the intermediate representation and our contribution to guideline quality are described.

For items where the AGREE instrument demands precise information about the guideline as a whole, such as for the first two domains, it is clear that any guideline representation must provide suitable slots where free text on these topics is store. However, our current focus is on representing the body of the guideline which seems more complex than storing a list of top-level guideline features.

Scope and purpose

This domain is concerned with the overall aim of the specific clinical questions and the target patient population.

The quality of the guideline is defined by the question, whether these three topics are specifically described in the guideline.

Stakeholder involvement

The focus here lies on the extent to which the guideline represents the views of its intended users.

This involves composition of the consortium, integration of patients' views, clear definition of target users of the guideline, and piloting of the guideline among end users.

Rigor of development

This relates to the process used to gather and synthesize the evidence, the methods to formulate the recommendations and to update them. To some of these items in this domain we can make contributions while others refer to using systematic methods for the search of evidence and describing them, plus external review of the guideline prior to publication.

Item 10. The methods used for formulating the recommendations are clearly described. This can be understood in two ways: Either "the methods are described on a general level", or/and "the basis for the formulation of each recommendation is clearly documented". Following the second interpretation, the links between statements in the intermediate representation can clearly improve the guideline quality. In the breast cancer treatment guideline, the summary recommendations do not contain explicit references to anything - neither literature nor literature summaries in the narrative.

Item 11. The health benefits, side effects and risks have been considered in formulating the recommendations. These aspects are modeled by our intermediate representation, compare Sections 1.3.6 and 1.3.7.

Item 12. There is an explicit link between the recommendations and the supporting evidence. In the guideline modeled in this project, this is the case for the evidence statements and parts of the

narrative, but not for the recommendations in the summaries. These links can be represented in our intermediate representation.

Item 14. A procedure for updating the guideline is provided. Here, our work to support living guidelines in general is the answer. However, issues of representation and editing of the guideline are only minor in this process – frequent consultation of domain experts is the major issue.

Clarity and presentation

This domain deals with the language and format of the guideline. While the formulation of the natural language sentences is on the border of the scope of our project, analyzing the content in the course of modeling it in intermediate representation, Asbru, and KIV should bring a major step forward in this domain.

Item 15. The recommendations are specific and unambiguous. and *Item 16. The different options for management of the condition are clearly presented.* aim at the core issue of the Protocure project.

Item 17. Key recommendations are easily identifiable While this is mainly a matter of text layout, any support from the content side should be provided by the intermediate representation.

Item 18. The guideline is supported with tools for application. The intermediate representation will not be an executable format. To arrive at an executable version of the guideline, it must be modeled in Asbru (or another guideline representation), which will be done in the course of this project. However, the execution environment used is for research purpose only and cannot be employed at the point of care. Software which can be used by medical personal in daily practice requires far more development resources than available in this project.

Applicability

This refers to organizational, behavioral, and costs implications of applying the guideline. While this again are features of the guideline as whole, the intermediate representation's attributes for cost and resources can contribute to a precise statement about these issues.

Item 19. The potential organizational barriers in applying the recommendations have been discussed. In part, this is difficult to cover in the intermediate representation, since it is not clear from looking at the guideline which recommendations will meet organizational barriers and which will not. In part, this can be covered by adding information concerning resources in the intermediate representation.

Item 20. The potential cost implication of applying the recommendations have been considered. From a modeling perspective, this is an easier version of the item before – every statement in the guideline can have a cost attribute in the intermediate representation. Comparing the cost of applying the guideline (which is precisely specified using the intermediate representation) with the cost of not applying it will be nearly impossible, but often the evidence for the benefit of a certain recommendation also includes information about the cost of not following it.

Item 21. The guideline presents key review criteria for monitoring and/or audit purposes. This refers to the guideline as a whole and is not covered by the intermediate representation. However, indicators can be modeled.

Editorial independence

The two items in this domain are concerned with the independence of the recommendations and the acknowledgment of possible conflicts of interest from the guideline development group.

1.2.4 Other Formal Representations of Guidelines

GLIF – Guideline Interchange Format

The authoring process. GLIF [7] supports representing guidelines in three levels of abstraction, marked by letters A to C. First, the guideline is authored at a conceptual level (A). This level enables the guideline author to concentrate on seeing the guideline as a flowchart without full details of decisions. The latter, together with patient data and iteration information are added to reach level B. Mapping actions to institutional procedures at a certain site and patient data references to a certain electronic patient data record leads to level C.

Representation of overall guideline features. *Author information* comprises authors, authoring date, guideline version, guideline status (published or not, obsolete), and developing institution of the free text guideline, plus a similar set of encoder, encoding last modification data, encoded guideline version, GLIF version used, representation status (of the syntax used), and adapting institution for the encoding of the free text version in GLIF.

Logic-oriented aspects describe intention, eligibility criteria, exceptions form the guideline, data items used by the guideline, and parameters passed to the guideline. Guidelines can be grouped into guideline collections within which one guideline can call another and exchange data with it. The data items are specified through ontologies which can be mapped to standard controlled vocabularies. Also, the activities described in a GLIF guideline can be mapped to a Reference Information Model such as that of HL7, which is also called Unified Service Action Model (USAM).

In addition, a list of supplemental didactic material can be given for each guideline.

Representation of guideline content. The guideline’s algorithm is composed of guidelines steps. These can be either action, decision, branch, synchronization, or patient state.

Action. An action step specifies a set of tasks to be performed. Attributes are iteration information, duration, triggering events, and associated exceptions. Actions can be nested. There are two types of actions: medically-oriented and programming-oriented. *Medically-oriented actions* specify a medical task as defined in the Reference Information Model (RIM) layer of GLIF’s data model. Examples include diagnostic test order, drug prescription, referral, and scheduling. *Programming-oriented actions* include sending messages, generating events, accessing patient data, assigning values to variables, and calling a subguideline.

Decisions. Decision steps direct the control flow between alternative steps. There are two varieties: case steps model deterministic decisions while choice steps model non-deterministic decisions. In the first case, the condition is (automatically) evaluated and the appropriate option is taken. If none of the options fit the data or if data is missing, a default action is chosen for continuation. In the case of choice steps, more than one option can be chosen. In this case, the user has to choose the option which is actually performed. The degree of preference can be modeled differently for different types of choices:

- A *rule-in* choice defines two criteria: a rule-in and a strict rule-in. The first is a condition which may hold for the choice to be applicable. The second is a condition which must hold.
- An array of criteria called *KofNChoices* where each criterion is associated with a weight. The sum of all fulfilled criteria of a choice determines how it is ranked on the list of choices.
- *Utility choices* use either a decision analysis tree or an influence diagram to compute the ranking of each option.

Branch. A branch step models concurrency of multiple guideline steps, which may be performed in parallel or in any order.

Synchronization. Synchronization steps are used in conjunction with branch steps. They mark the place where the different branches of execution meet again and specify the conditions to proceed, i.e., whether all, some or one of the preceding steps must have been completed before continuing.

Patient state. A patient state step labels its position in the guideline for two purposes. On the one hand it shows the progress of the patient state. On the other hand, it serves as an entry point of the guideline. This means that guidelines can be started at any place which contains a patient state.

GEM – the Guideline Elements Model

The GEM project consists of the Guideline Elements Model itself, the editing tool GEM-Cutter, and the quality evaluation method GEM-Q.

The authoring process. GEM [11] was developed by the Yale Center for Medical Informatics as a means for the implementation of existing guidelines in a concrete care setup. This implementation takes place in three steps. First, the GEM document is created based on the original guideline using the GEM Cutter. GEM is a XML-based syntax. The GEM Cutter resembles the GMT, but it does not maintain links between fractions of the original text and the corresponding GEM elements. The GEM document can be freely edited and no book keeping of these changes is implemented, i.e., there is no measure for the authenticity of the GEM document. The elements of the GEM document are then stored in a relational *design database*.

In a second step – named knowledge customization – meta-information is added. This process is guided by a program called *knowledge customization wizard*. Examples of the added information are details on data input for decision variables and adaptation of actions and directives to the model of actions used in HL7. Besides adding meta-information, this step also comprises local adaptation of the guideline and concrete implementation of abstract concepts in the guideline.

The third step in the guideline implementation process is *Knowledge Integration* into the clinical workflow depending on the local circumstances.

Representation of overall guideline features. GEM provides about 100 different XML elements to represent different features of a guideline. The majority of them describes properties of the guideline as a whole, e.g., title, developer, purpose, target population, method of development. Some of these fields use controlled vocabulary from National Guideline Clearinghouse, e.g., for intended users or clinical specialty.

Representation of guideline content. There are three groups of elements describing the content of the guideline in detail: Recommendations, definitions, and algorithm.

Recommendations can be conditional or imperative. For conditionals, the decision variable and its value are given (as free text) together with (optional) sensitivity, specificity, predictive value, and cost of testing the decision variable. For the recommended action benefit, risk, and cost are stored. In addition, for each recommendation reason, evidence quality, strength of recommendation, flexibility (optional actions), cost, etc. are annotated.

Definitions simply consist of two parts: Term and term meaning. Both are free text.

Algorithm consists of 4 different elements related to GLIF, namely action step, conditional step, branch step, and synchronization step. GLIF's decisions is marked up as conditional recommendation in GEM.

ProForma

ProForma [5] is aimed at the development and/or implementation of guidelines and protocols using a graphic editor. The design process consists of two steps:

1. Developing a high level diagram which describes the outline of the guideline in terms of four different types of tasks: plan, decision, action, and enquiry.
2. Converting this graphical structure into a database and populating it, using software implementation of the task templates.

Plans contain other plans, decisions, actions, or enquiries. Actions represent treatment steps. Enquiries represent data input. ProForma features an elaborate decision process. Decisions list a set of argument. Each argument contains a condition and refers to a particular option. It has one of four modes: *for*, *against*, *confirming*, and *excluding*. If there is both a confirming and an excluding argument for an option then this case is undecidable. Otherwise, a single confirming argument overrules all arguments against the option. Similarly, a single excluding argument overrules any arguments for the option. If neither confirming nor excluding argument, then the majority among the arguments for and against the option wins.

Glare

The developers of Glare [13] distinguish between epistemological and ontological knowledge. The first is domain independent and describes the basic types of entities such as atomic and composite actions, the structural relations between entities, and the control relations between entities. The ontology describes the basic attributes of the entities in a given domain (such as medicine) and at some of the basic and most frequently recurring entities in the domain (e.g., diagnosis).

Basic entities are actions in the broad sense, similar to other approaches. Actions can be composite (consisting of other actions) or atomic. There are four types of atomic actions: work actions, query actions, conclusions, and decisions. *Work actions* refer to activities of care personal, similar to user-performed plans in Asbru or actions in ProForma. *Query actions* request information as does the *ask*-statement in Asbru or enquiries in ProForma. *Conclusions* are the different outcomes of a decision process and resemble the choices in GLIF or ProForma. *Decisions* are similarly modeled as in GLIF and ProForma.

Structural relations of actions form a hierarchic tree similar to the hierarchy of plans in Asbru.

Control relations define the order in which actions are performed: Sequential, concurrent, alternative, or repetition. Concurrent actions correspond to unordered plans in Asbru. For alternative actions, only one of a set is performed.

Attributes of work actions. On the ontological level, different work actions are distinguished, namely clinical procedure and pharmacological prescription. For clinical procedures, the attribute groups are preconditions, goals, and cycles. Preconditions comprise "may" and "must" inclusion and exclusion criteria like GLIF and ProForma, a description of potential conflicts within these rules, and the cost, time, and resources associated with this task. Goals are describes as free text. Cycles are described by frame time (duration), granularity (e.g., hour or day), grouping (e.g., every 4 hours), repetition (e.g., 3 times), execution time, delay between iterations, and exit condition. Pharmacological prescriptions are represented by the name and description of the drug.

Attributes of query actions are time and cost to obtain the desired information.

Attributes of decision actions. Here, a distinction is made between diagnostic and therapeutic decisions. While the first ones are described to be very complex, the second are described by effectiveness, cost, side-effects, compliance, and duration for each of the treatment options.

Degel – Digital Electronic Guideline Library

Degel [10] focuses on the problem that on the one hand guidelines which are only free text are inaccessible to health care staff and on the other hand it would require unrealistically huge effort to translate all of today’s guidelines completely into a formal representation.

As a compromise, the gradual migration of free text guidelines to formal representation is supported by a generic framework with tools to support guideline classification, semantic markup, context-sensitive search, browsing, run-time application, and retrospective quality assessment.

Degel is ”DTD-driven”. This means that it is applicable for any XML-based guideline representation. Current implementations support Asbru and GLIF.

Guideline classification along different semantic axes is performed using the IndexiGuide tool. Currently used axes are: symptoms and signs, diagnostic findings, disorders, treatments, body systems and regions, guideline types, and guideline specialties.

Semantic markup is performed using the Uruz web-based guideline markup-tool, which resembles the GMT but does not maintain links between different representations of the guideline. Several features are especially tailored to Asbru, such as the *plan-body wizard*.

Context-sensitive search and retrieval is performed by the Vaidurya tool. The user can select concepts from the semantic indexing axes together with marked-up terms in formal versions of guidelines.

Browsing and Visualization of guidelines is performed by the VisiGuide tool.

In addition, the Spock run-time application module and the Asbru-specific QualiGuide retrospective quality assessment tool are under development.

Stepper

Stepper [12] is a tool similar to GMT to transform a narrative guideline into a formal representation in several steps. Each step is performed semi-automatically. The translation steps are described in XKBT, which extends XSL by interactive input. Stepper is the only system besides GMT which maintains links between the different levels of refinement.

In a first step, the basic blocks of text are marked (e.g., headings, sentences) and parts without operation semantics are removed.

In a second step, complex sentences are rearranged into simpler ones and background knowledge is added. In addition, a data dictionary is created, which describes the clinical parameters involved.

In a third step, the original document is transformed into a *universal knowledge base*. This involves changing the structure of the document to achieve modularity, which is assumed to involve medical experts in part.

In a fourth step, the representation is adapted to ease the export to the target representation. Therefore, an export-specific knowledge base is produced from the universal one.

Finally, the ultimate format is produced by a knowledge engineer. This final step is assumed to be performed fully automatically using XSL style sheets. Examples for final formats are rule bases in Prolog or class structures in Java.

Our guideline modeling process integrates the first two step in one, which is the creation of the intermediate representation from the original guideline.³ Optionally, some standard separation of the guideline into chunks (sentences) can be performed as preprocessing. This is suitable for long guidelines with large pieces of narrative text. The third step of Stepper resembles our second step, in which a knowledge engineer ensures the completeness of information and acquires missing information. The pendant to the fourth step of Stepper is the translation of the intermediate representation to Asbru. In the Protocure project, Asbru is translated to KIV which can be

³No changes of the sentences of the original guideline are performed.

seen as the ultimate format in Stepper terminology. However, for guideline execution this step is not necessary since the Asbru code is interpreted by the execution unit without any further transformation.

1.3 Details on the Intermediate Representation – The Many-Headed Bridge

Several teams have tried to bridge the gap between a natural language guideline and its formal, computer executable representation. Some have built multi-part bridges reaching the other shore in several steps, jumping from isle to isle. Our vision is to connect all these isles by one bridge with more than two heads. We therefore call it the *many-headed bridge*. It is abbreviated MHB.

Design principles

Structure based on natural language text. The overall structure of an MHB file is very flexible. It is a series of *chunks*. Each chunk corresponds to a certain bit of information in the natural language guideline text, e.g., a sentence, part of a sentence, or more than one sentence. Initially, the order of the chunks reflects the order of the text they refer to in the English version of the guideline. However, they can be moved freely in the MHB file. Such regrouping is optional and may provide advantages as a preparation for constructing a more formal representation, e.g. in Asbru, based on the MHB representation.

Triple purpose. MHB is designed to serve for three purposes.

1. As a bridge between a single living guideline and its Asbru representation or another formal guideline representation.
2. As a means to improve the quality of a guideline through the detailed modeling of the evidence base for each recommendation in the guideline.
3. As a utility for the merging of two or more guidelines on the same topic.

The latter two will not be explored in the Protocure project but both of them are considered in the design process. Table 1.1 shows which of the aspects represented in MHB are important to bridge the gap to Asbru or other formal guideline representations and which are intended for quality assurance of the guideline without using a formal representation such as Asbru.

MHB is not designed as the *only* representation for the design of a new guideline from scratch. However, assuming that the design of a new guideline will immediately involve draft pieces of natural language text, it can be considered to fall into the first category – evolution of a living guideline.

Independence of other representations. MHB is designed as a bridge between any natural language and any formal guideline representations. While only Asbru is used in the Protocure project, the features of GEM, GLIF, Proforma, and Glare are also covered. Section 1.2.3 discusses the relation between the work described in this document and the AGREE instrument.

MHB must not be seen as an interlingua between formal guideline representations. It is an *intermediate* representation which is far less formal than Asbru, GLIF, etc. The latter specify every detail in precise enough for automatic processing while MHB is not aimed at automatic execution or verification and therefore does not provide this level of detail in the syntax.

Flexible syntax. MHB is meant to be an *intermediate* representation to bridge nearly unstructured text to formal guideline representations. Therefore, a strict definition of the content of each field on a *syntactical* level would impose too much restraints in the modeling process. On the other hand, we provide strong *semantical* definitions of the content of each field, ensuring consistent interpretation of a file by different persons.

Dimension or Aspect	Bridge to Asbru	Quality Improvement without Asbru
Control Flow	●	
Data Flow	●	
Temporal Dimension	●	
Evidence		●
Background		
- Intention & Effects	●	●
- Educational Information & Indicators		○
Resources	○	○
Patient aspects	○	○
Structure	●	○

- important
- useful

Table 1.1: Importance of different dimensions of a chunk in MHB as a basis for the creation of a formal representation in Asbru (middle column) and for (future) quality improvement using MHB independent of another formal representation (right column).

Control flow, *data flow*, and *temporal dimension* together with intentions and effects provide the foundation for building a formal model of the guideline, e.g., in Asbru.

Clear statements of *evidence*, *intentions*, and *effects* are already important features of good quality guidelines.

Detailed descriptions of *resources* needed to implement the guideline and of the related *patient aspects* improve the quality of the guideline even further.

Indicators should become part of future guidelines, currently they are defined separately.

The content of *educational information* is difficult to model, but certainly important for the users of the guideline.

The guideline *structure* forms an important background for the interpretation of each piece of information. E.g., there is a difference in the weight of evidence summary statements and introduction.

1 Element **root**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
zero or more			
chunk	#3, p. 19	zero or more	One bit of information
chunk-group	#2, p. 18	zero or more	Used to structure the file
refer-to	#6, p. 20	zero or more	Reference to another chunk-group

2 Element **chunk-group**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT. Pointing to title of section.
zero or more			
chunk	#3, p. 19	zero or more	A single bit of information
chunk-group	#2, p. 18	zero or more	Another group nested in this group
refer-to	#6, p. 20	zero or more	Reference to another chunk-group or chunk or title of external document
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
title	CDATA	optional	Section name or other internal categorization
group-id	CDATA	optional	Unique ID used to jump to refer to this group in refer-to

This element is used in: *root* (#1, p. 18), *chunk-group* (#2, p. 18)

Independent groups of aspects. The pieces of information about each chunk are called the *aspects* of a chunk in MHB. These aspects are grouped in *dimensions*. The aspects of different dimensions are independent but sometimes related. Every aspect is optional. In practice, for most chunks only aspects of a few dimensions are given. The reason for this lies in the fact that chunks are a uniform representation for all the heterogeneous parts of the guideline.

The high flexibility of MHB avoids representation problems which are unavoidable with more rigid representations, but it makes the task of checking completeness and consistency of the MHB file rather complex. However, using XSL and the GMT, this task can be solved. The use of XSL in GMT is explained in deliverable D2.3a.

The basic structure of an MHB file

A file in MHB is a series of chunks. Each **chunk** element has one optional child element for each of the dimensions. Each of them contains a series of optional elements storing the aspects of that dimension. The remaining subsections describe the representation of each of these dimensions.

Each element representing aspects of a dimension contains optionally a **link** element. This again contains an ID referring to a part of the natural language text. The ID has a prefix to distinguish different natural language texts, which is important for merging guidelines. The link element may contain one or more **source** elements. Each of them contains a **source-file-id** and **source-text**. The part of the natural language text to which the link refers *can* be copied into these attributes in the MHB file. While this is useful for merging several existent guidelines, the use of **source** is discouraged for the maintenance of a living guideline because changes in the original text do not automatically propagate to the corresponding **source-text** attribute in the

3 Element **chunk**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
zero or more			
control	#7, p. 21	zero or more	Aspects in the control flow dimension
data	#15, p. 27	zero or more	Aspects in the data flow dimension
time	#20, p. 28	zero or more	Aspects in the temporal dimension
evidence	#23, p. 31	zero or more	Aspects in the evidence dimension
background	#24, p. 33	zero or more	Aspects in the background dimension
resources	#31, p. 35	zero or more	Aspects in the resource dimension
patient-aspects	#32, p. 36	zero or more	Aspects in the patient dimension
structure	#33, p. 36	zero or more	The status of the chunk in the original guideline text

<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
chunk-id	CDATA	required	used to refer to by other chunks
subject	CDATA	optional	Name or description of action or event which is described here, if not clear from the context.
context	CDATA	optional	Specification of context for this chunk, if not clear otherwise.

This element is used in: *root* (#1, p. 18), *chunk-group* (#2, p. 18)

4 Element **gmt-link**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
source	#5, p. 20	optional	Added by tool for guideline merging purposes

<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
link-id	CDATA	required	ID issued by GMT

This element is used in: *refer-to* (#6, p. 20), *chunk-group* (#2, p. 18), *if-then* (#8, p. 23), *decomposition* (#10, p. 24), *synchronization* (#12, p. 25), *repetition* (#13, p. 25), *clinical-activity* (#14, p. 26), *definition* (#16, p. 27), *usage* (#17, p. 27), *input* (#18, p. 27), *abstraction* (#19, p. 28), *start* (#21, p. 29), *qualitative-relation* (#22, p. 29), *evidence* (#23, p. 31), *educational* (#28, p. 34), *explanation* (#29, p. 34), *indicator* (#30, p. 35), *intention* (#25, p. 33), *effect* (#26, p. 34), *relation* (#27, p. 34), *resources* (#31, p. 35), *patient-aspects* (#32, p. 36), *structure* (#33, p. 36)

5 Element **source**

<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
source-ID	CDATA	required	Code for original file or guideline version
source-text	CDATA	required	Text snippet to which the link points

This element is used in: *gmt-link* (#4, p. 19)

6 Element **refer-to**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT. Pointing to title of section.

<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
target	CDATA	required	group-id of the target chunk-group

This element is used in: *root* (#1, p. 18), *chunk-group* (#2, p. 18)

MHB file.

In some cases, it is desirable to attach a label to the chunk specifying the context of what is described by the content of this junk. This label is stored in the optional attribute **subject**. An example would be a certain treatment about which several distinct things are said in one paragraph. If this paragraph is modeled as several junks, it might be desirable to specify the (short) name of the treatment in each junk.

Guidelines contain cross references to other guidelines or other sections of the same guideline. These cases are handled by the **refer-to** element. For references to other guidelines its attribute **target** contains the name of the guideline. For references to other parts of the same guideline, **target** contains either the ID of the chunk-group or the chunk referred to, whichever is appropriate.

1.3.1 Control flow

One of the most prominent aspects of a guideline is, *when* to do *what*. This is often shown (incompletely) by a drawing called clinical algorithm. *When* either relates to the condition, under which something happens, or to the temporal order of actions relative to each other. The first is generally called *decision*, while the latter can be subsumed under *ordering* of activities and/or events. A third issue is *decomposition*, i.e., the way in which larger tasks or activities are made up of smaller ones. Last not least, some tasks are performed *repeatedly* as part of other tasks.

Ordering, decomposition, and repetition imply temporal aspects, e.g., one task being performed after the other. Still, they are included here, while the dimension *temporal aspects* deals with explicit and often quantitative descriptions of the timing of actions and also effects etc.

Decisions

The prototypical form of a decisions is "if condition then result". However, there are many details to consider.

- *Result* can be a event, an effect, or an activity but in this section we only refer to activities. An activity has a personal or personalizable actor, e.g., the physician on duty, whereas a event does not have a person as its actor or driving force, e.g., a disease. For instance, in the event of Jaundice, the actors are the Bilirubin molecules or the not yet sufficiently working liver. There is an important impact of this distinction: The events of the disease are not

7 Element **control**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
zero or more			
if-then	#8, p. 23	zero or more	A pair of condition and result plus modifiers for each
option-group	#9, p. 23	zero or more	A set of related condition/result-pairs
decomposition	#10, p. 24	zero or more	A parent task and its children
synchronization	#12, p. 25	zero or more	Information of the synchronization of tasks
repetition	#13, p. 25	zero or more	A parent task and a repeated child task
clinical-activity	#14, p. 26	zero or more	An activity for which only a description is given.

This element is used in: *chunk* (#3, p. 19)

synchronized with the activities of the care process. Therefore, modeling both as a single hierarchy of plans is rarely feasible. Usually, only the care activities are modeled and the event of the disease itself becomes visible through the result of diagnosis steps only. This resembles the view of the physician.

Effects are changes in some observable parameters where nothing about the underlying process is known – for instance, "X leads to decreased mortality". The rate of mortality can be observed, but everything between the (assumed) cause X and the resulting rate change is unknown. Effects are described by the dimension *background*.

- Often a decision has many options or results depending on several inputs. E.g., age plus Bilirubin level lead to one of four treatment options. This is technically the same as a set of decisions, but it is important for human understanding to use appropriate forms for complex decisions as discussed below.
- In the medical domain, few statements are imperative. Most statements contain some form of vagueness or restriction. This can be qualitative such as the word "should" or it can be quantitative such as "in 70 % of all cases". All these aspects are subsumed in the dimension *evidence*. Still, they are directly embedded into the decision statement in some formal representations, which is why references to them appear below.

From the formal modeling point of view, the main distinction is made between mutual exclusive and non-excluding options. In the first case, clear directives how to arrive at exactly one of the given options are specified, be it in a decision tree, a decision table, or simple if-then rules. In the second case, for each of the options there are one or more reasons to take it or not to take it and there is no a-priori safety that exactly one option will be taken.

Both, Proforma and GLIF distinguish between rule-in (reasons to take a certain option) and rule-out (reasons not to take a certain option) standard and strict rules. Strict rules overrule standard rules. In both Proforma and GLIF each rule can receive its own weight. In contrast, Asbru forces the guideline developer to detail how to combine the various conditions to do or not to do something.

In MHB the basic structure of a decision is **if-then**. It consists of a **condition**, a **condition-modifier**, a **result**, and a **result-modifier**. All four are attributes formally containing any text. Semantically, the condition is described as precisely as possible based on the guideline text. It should

– if possible – contain concepts described in the data dimension. The `condition-modifier` is versatile to support various categories of choices:

- For simple recommendations, `condition-modifier` remains empty.
- For rule-in/rule-out choices, `condition-modifier` takes one of the four values `strict-rule-in`, `rule-in`, `rule-out`, `strict-rule-out`. It lies in the responsibility of the author to follow this consistently. A stricter syntax definition of MHB would hinder the gradual migration from more vague concepts to this strict scheme.
- The words *should*, *can*, etc. which some times are found in recommendations are stored in `condition-modifier` alternatively to the above.
- If the author of the MHB file acquires numerical weights either in the guideline or from domain experts, these can be stored in the `condition-modifier` instead of the qualitative labels like `rule-in` etc.
- For negative recommendations, the word *not* is entered as `condition-modifier`. Alternatively, the precise words of the guideline can be copied to `condition-modifier`, e.g., to make the distinction between "not recommended" and "recommended not to do" where desirable.

The `result` designates the recommended action, first in the words of the original guideline, latter using a name from the plan or task hierarchy gradually build in the process of revising the MHB version of the guideline. If omitted, `result-modifier` defaults to simply *do* but it can also take values such as *start*, *stop*, *suspend*, etc. These depend on the target representation and on the precision of the information in the guideline. The usage of the `result-modifiers` *abort* and *complete* is only useful if the target formal representation is Asbru because other representations do not distinguish between successful completion and failure of a plan or task.

Sometimes more then one recommendation or option are described together. In MHB the element `option-group` is used to group several `if-then` elements. The options can exclude each other or not. The attribute `selection-type` has the values `single-choice` or `multiple-choice` to represent this distinction. The additional (optional) attribute `other-selection-specification` can contain more complex constraints on the selection, such as the number of options to select together, e.g., "perform two of the following five tasks".

Ordering and decomposition

A parent task (or plan) *P* can be *decomposed* into children *A*, *B*, and *C*. *Ordering* defines constraints on their execution relative to each other. Both ordering and decomposition are modeled by the same element in MHB named *decomposition*.

Several formal approaches pursue different models of the ordering of tasks, e.g., flowchart, task network, or hierarchy of plans. Each of them can be mapped to each other.

Some representations, such as Asbru, have constructs for frequent special cases for constraint combinations, such as *sequential* or *any order* (which means that only one plan may be active at a time). A representation which suits any possible case must be able to constrain both start and end of each task or plan based on any other, and constraints for start and end must be independent. For instance, *parallel* most of the time means that plans start at the same time (but often some tolerance is allowed). But is rarely defined whether they must end together and what the consequences of a violation of this rule are.

While the form of presentation is only a formal problem, the lack of information about the precise constraints of ordering in the guideline can be an important issue. The examination of such information gaps can contribute to improved quality in guideline design even though many gaps will be intentional.

The MHB element representing these relations is `decomposition`. It names a parent plan or task as an attribute and the names of child plans in separate elements. This is necessary because

8 Element **if-then**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
degree-of-certainty	CDATA	optional	Put words like may or sometimes here
condition	CDATA	required	Precise description of condition, preferable in terms of variables defined by data definitions
condition-modifier	CDATA	optional	(Strong) rule-in/out for GLIF/ProForma-style; or not for negative recommendations; or weight for weighted rules; or empty for plain conditions
result	CDATA	required	What to do if condition fulfilled, preferably in terms of a task name described somewhere else
result-modifier	CDATA	optional	Start/stop for language independence; or activate/abort/complete for Asbru Light; or empty for 'just do it'

This element is used in: *control* (#7, p. 21), *option-group* (#9, p. 23)

9 Element **option-group**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
if-then	#8, p. 23	one or more	A pair of condition and result plus modifiers for each
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
parent-task	CDATA	optional	Unique task name of parent task
selection-type	single-choice	required	single-choice: only one choice is taken, multiple-choice: choices do not exclude each other
other-selection-specification	CDATA	optional	Number of options to select or comments on the selection process

This element is used in: *control* (#7, p. 21)

10 Element **decomposition**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
child-task	#11, p. 24	one or more	Each element represents one child task.

<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
parent-task	CDATA	required	Unique task name of parent task
task-description	CDATA	optional	Details of the task which are not formalized.
ordering	CDATA	optional	Sequential or parallel etc.
order-child-constraints	CDATA	optional	Select one etc.
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *control* (#7, p. 21)

11 Element **child-task**

Aliases: **awaited-subtask**

<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
name	CDATA	required	Unique child task name
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

child-task is used in: *decomposition* (#10, p. 24)

awaited-subtask is used in: *synchronization* (#12, p. 25)

of a limitation of XML: Several attributes with equal names are not permitted in a single XML element.

The ordering of the children is specified in attribute **ordering** of element **decomposition**. Its values should map to plan orderings available in the targeted formal representation, e.g., *any-order* can only be implemented in Asbru. If the ordering is not given in the guideline, then this attribute is omitted.

Synchronization

When several tasks are performed in parallel or otherwise independent from each other, the question arises when to pursue the rest of the guideline.

Asbru and GLIF define those subtasks ("children", **awaited-subtasks** in MHB) which must be completed before the next step is taken in a logical expression. In both cases, the number of completed children can be given alternatively.

An important side question is whether child tasks which are not completed are terminated when the main task is resumed or not. This is treated different in different approaches. Again, clarifying this point can improve the quality of a guideline, or it can be an exercise necessary for the formalization but not yielding additional insight to readers of the guideline.

The issue of continuing child tasks the end of which is not awaited is treated different in different formal guideline representations and information about it in guidelines is sparse. MHB therefore merely contains an optional attribute **continue-other-subtasks** for remarks on this issue.

12 Element **synchronization**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
awaited-subtask	#11, p. 24	one or more	Each element represents on task to be waited for.
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
waiting-task	CDATA	required	Name of task, which waits. In Asbru this is the parent plan.
continue-other-subtasks	CDATA	optional	Are there child tasks which are continued after synchronization?
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *control* (#7, p. 21)

13 Element **repetition**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
envelope-task	CDATA	required	Unique task name of parent task
repeated-task	CDATA	required	Unique task name of child task
repeat-specification	CDATA	required	How often and with which constraints is the child repeated?
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *control* (#7, p. 21)

Repetition

Often one task is performed more than once – either for a certain number of times or at certain times during another task. In any case, there is a parent plan (“envelope task”) which lasts during all the repetitions and a child task (“repeated task”) which is performed repeatedly. The form of repetition together with all its constraints is entered in the attribute `repeat-specification`.

Atomic actions

Some activities or tasks are only described in a sentence without further detailing it. In these cases, they are generally called *atomic actions* which may be too strict if taken literally. Therefore, the MHB element to model such cases is called `clinical-activity`.

1.3.2 Data flow

Interwoven with control flow is the description of the data processing involved in the diagnosis and treatment of the patient. While the processing of data seems of little importance in the treatment of many diseases, it is often prominently described in the diagnosis part of a guideline. As control

14 Element **clinical-activity**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
name	CDATA	required	Unique task name
description	CDATA	required	Details on the activity.

This element is used in: *control* (#7, p. 21)

flow describes the gathering of information, data flow describes how one piece of information is abstracted from other ones. We distinguish the following:

- The *definition* of a data item is rarely found in the guideline in explicit form. Still, it is necessary for the formal version of the guideline. It consists of a name, a type, and often a range of plausible values and a preferred unit.
- The *usage* of a data item is made explicit to varying degrees in actions described in the guideline and calculation of other values.
- The *input* of a data item is sometimes explicitly described in the description of the patient interview or diagnosis.
- *Abstraction rules* describe the calculation or abstraction of one data item based on others. It is found mostly in descriptions of diagnosis (compare example 4). The time at which the calculation or abstraction is performed may be explicitly stated or not. In the first case, the statement in question has a data flow aspect and a control flow aspect at the same time. In the second case, abstraction is assumed to take place automatically whenever necessary.

Example 4 *Locally spread marma carcinoma means a marma carcinom which is irresectible based on the classic irresectibility criteria: (list of different types of carcimomas). Big primary tumors (> 5cm, class T3) fall into this category.* p. 64, definition

All guideline modeling approaches have means to define, input, and compute pieces of data or information. Only the Asgaard framework has a data abstraction unit which automatically abstracts values based on the current input.

Various formal representations offer means to make the binding of variable names to concepts in an Electronic Medical Record or an ontology or a controlled vocabulary explicit. Asbru does not. Research in this direction is not part of Protocure II.

Similar to representations such as GEM, the definition of a data concept (i.e., variable) simply consists of two strings, one for the name of the concept and the other for its description.

The usage of a concept is marked by element **usage** which only contains the name of the concept or variable. The same holds for input. Data abstraction rules are defined in element **abstraction** which contains the name of the result of the abstraction and the abstraction rule. As for conditions, the abstraction rule should be as close to the syntax of the target representation to reduce problems in the conversion of MHB to that representation, but the syntax definition of MHB does not enforce any details to allow gradual migration towards precision.

1.3.3 Temporal aspects

Both data and control flow may have temporal aspects. They can be qualitative or quantitative. Qualitative temporal relations between time points are *before*, *after*, and *equal*. For two intervals, Allen defined 13 relations based on all combinations of the relations of time points. While they are well known, they are a rephrasing of the relations of time points.

15 Element **data**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
zero or more			
definition	#16, p. 27	zero or more	Definition of one data entity, i.e., variable
usage	#17, p. 27	zero or more	Names a variable or piece of information which is used in this chunk
input	#18, p. 27	zero or more	Designates a place where a certain bit of information is supplied by the user
abstraction	#19, p. 28	zero or more	Describes how one variable is calculated based on others in a declarative way

This element is used in: *chunk* (#3, p. 19)

16 Element **definition**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
name	CDATA	required	Unique variable name
description	CDATA	required	Semantics or link to patient record
technical-specification	CDATA	optional	Data type, value range, initial value, etc.

This element is used in: *data* (#15, p. 27)

17 Element **usage**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
name	CDATA	required	Unique variable name of the used value
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *data* (#15, p. 27)

18 Element **input**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
name	CDATA	required	Unique variable name for input
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *data* (#15, p. 27)

19 Element **abstraction**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
result	CDATA	required	Unique variable name
abstraction-rule	CDATA	required	Precise description of calculation
description	CDATA	optional	Semantics or other information

This element is used in: *data* (#15, p. 27)

20 Element **time**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
zero or more			
start	#21, p. 29	zero or more	Starting time of an interval
end	#21, p. 29	zero or more	Finishing time of an interval
duration	#21, p. 29	zero or more	Duration of an interval
qualitative-relation	#22, p. 29	zero or more	Qualitative temporal relation
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
subject	CDATA	optional	Name or description of action or event which is described here, if not clear from the chunk subject.

This element is used in: *chunk* (#3, p. 19)

Quantitative temporal relations define the temporal distance (i.e., time) between two time points. Often, such information is not given as a single figure, but as a pair of minimum and maximum. Sometimes only one of both is known.

While standard Asbru is very strong in modeling temporal relations including uncertainty and incompleteness, we excluded temporal aspects largely from Asbru light in Protocure I to ease its modeling in KIV, and because there was very little information about temporal aspects in the guidelines we modeled.

MHB covers the complexity of Asbru but adds more standard concepts. As for other cases, there is no pressure for completeness.

For each of start, end, and duration, the minimum, maximum, estimate, and precise value can be given. Of course the precise value excludes others, but the other three values can be combined, i.e., minimum, maximum, and estimate can be given together, if ever found in a guideline. The difference between estimate and precise value lies in the semantic given in the guideline. If start or end are given relative to a certain starting point and it is not obviously the start of the plan described, then reference point must be noted together with the offset in the respective attribute.

In addition to the above, the temporal dimension also models qualitative temporal relations such as "A is started after the end of B". While this could be implemented using the above elements, we provide a distinct element for qualitative relations to improve comprehensibility of the model.

21 Element **start**

Aliases: **end**, **duration**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
reference-point	CDATA	optional	Reference point for relative expressions in other attributes
minimum	CDATA	optional	Lower limit (absolute or relative to reference-point)
maximum	CDATA	optional	Upper limit (absolute or relative to reference-point)
estimate	CDATA	optional	General approximation given in the guideline or by a domain expert
precise-value	CDATA	optional	Exact value if known
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

start is used in: *time* (#20, p. 28)

end is used in: *time* (#20, p. 28)

duration is used in: *time* (#20, p. 28)

22 Element **qualitative-relation**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
left-hand-side	CDATA	required	Right-hand side of relation, e.g., A.
relational-operator	CDATA	required	Operator relating left-hand side to right-hand side, e.g, after.
right-hand-side	CDATA	required	Right-hand side of relation, e.g., B.
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *time* (#20, p. 28)

1.3.4 Evidence

An evidence-based guideline builds a bridge from carefully examined pieces of evidence which are obtained for certain particular parts of the problem to generally applicable recommendations for diagnosis, treatment, screening, etc. While it might be an interesting task to document the foundation of each an every sentence in the guideline, it will be too tiresome in practice.

Evidence for a statement is seen in various forms:

- For statements in the designated list of evidence-based recommendations, a grade of evidence is given. In addition, the literature references given for this statement are grouped by grade of evidence each of them provides.
- Statements in the guideline can have a literature reference.
- The literature references are described in an evidence table.
- Some statements explicitly refer to other parts of the guideline by phrases such as "as describe in Section ...".
- Many statements refer to previous statements without explicitly stating this. Such relations are detected by the common-sense of the reader. However, the limit to which such relation can be seen in a text is not easy to draw.

For the *explicit* references with defined format (statements of evidence, literature references) MHB provides the attributes **grade**, **level**, *importance* and **literature-reference**. They are all optional and can be combined as suitable. However, only some of the many theoretically possible combinations are used in practice.

- Literature references in the scientific explanation are not rated or graded, they are only represented by storing the reference (number or name of first author) in **literature-reference**.
- Literature references in evidence conclusions have a **level** which is stored in the attribute of this name while the reference is again stored in **literature-reference**.
- Evidence conclusions have a **grade**.
- Evidence conclusions and/or recommendations may have an importance attached to them by the guideline authors independent of the grade of evidence. In these cases, the attribute **importance** is used in parallel to **grade**.

For *implicit* references in the guideline, it can help to improve the quality of the guideline to make them explicit in the MHB file. The attribute **is-based-on** contains a reference to another MHB element. The ID used is that of the chunk on which the claim in this chunk is based on.

In practice, making implicit links explicit will be limited by a trade-off between the work investment into this task and the expected gain in quality.

In GLIF, each guideline step has a strength of evidence and action steps (i.e., recommendations) have a strength of recommendation in addition. Each of them consists of a string describing the coding scheme and another describing the actual strength. Strength of *evidence* marks the way the guideline authors evaluate the guideline, while strength of *recommendation* indicates how strong the authors urge the reader to follow this recommendation. In addition, evidence can be provided through supplemental material as described in the next section.

In Asbru, the evidence can be coded as a comment or an explanation which is shown to the user when a user-performed plan is executed, which is the encoding of a recommendation in the guideline.

Figure 1.1 shows an example for an evidence-based conclusion on a CBO guideline.

23 Element **evidence**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
is-based-on	CDATA	optional	ID of other chunk
literature-reference	CDATA	optional	Citation of article
grade	CDATA	optional	Grade of evidence-based conclusion according grading scheme of guideline
level	CDATA	optional	Level of evidence in the literature according grading scheme of guideline
importance	CDATA	optional	Importance of statement rated by guideline authors

This element is used in: *chunk* (#3, p. 19)

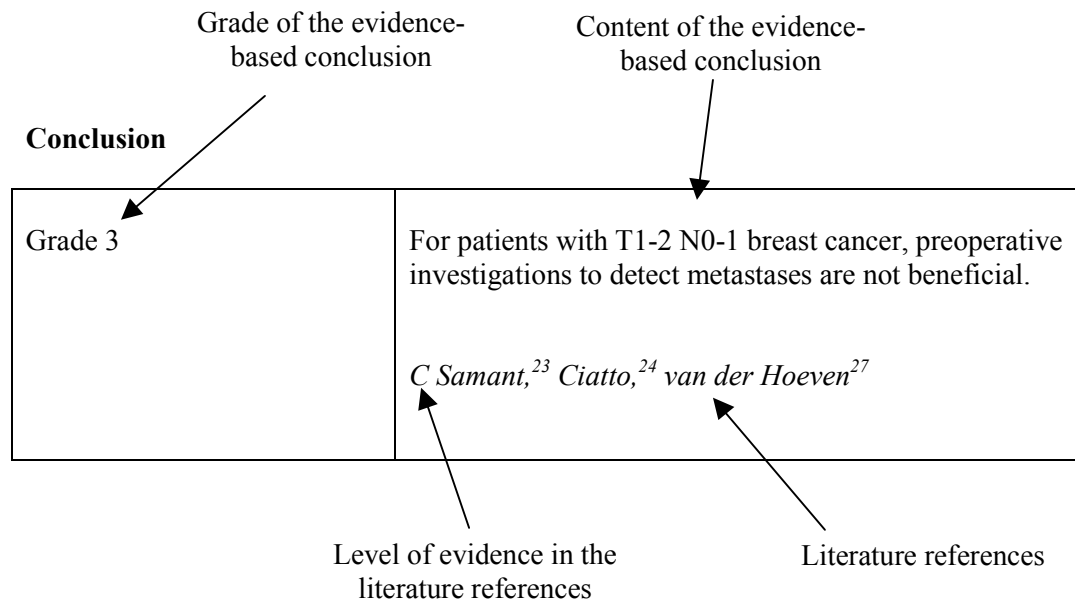


Figure 1.1: Sample evidence-based conclusion statement.

1.3.5 Background information

Background information describes various aspects of the topic. Some refer to a particular statement or group of statement while others are only loosely coupled to particular statements or recommendations. Also their potential to be formally encoded largely varies.

- *Intentions* of the described actions or recommendations are important in the Protocure project as they provide input to the proof process. They also inform and motivate the reader about the reason of certain steps as in example 5.
- *Effects* are relations between data or phenomena and other phenomena which are not seen as events or actions. They also play an important role in the proof process. Example 6 shows a statement which describes an effect which at the same time can be seen as intention or motivation for this kind of therapy.
- *Relations* are similar to effects, but do not postulate that one of the two named entities is the cause of the other.
- Other *educational information* can be targeted at the physician or the patient to discuss the recommendations in detail. To some extent, it is related to evidence, as it is an alternative backing for the recommendations in the guideline. Still, there is a considerable amount of statements in the guideline, which merely contain important information without direct link to any activities, such as that in example 7.
- *Explanations* are an important subspecies of the above item. They contain information directly explaining recommendations or other (important) statements in the guideline. In an implementation of the guideline, it might be useful to show them to the user (patient or physician) – in contrast to the education information above which is not as directly related to the actions resulting from following the guideline.
- *Indicators* define measures to assess the degree to which a guideline is followed in practice or the success of following this guideline. They are not yet part of guidelines (at CBO) but will be integrated in the future whenever possible. Including indicators relates to AGREE item 21: "The guideline presents key review criteria for monitoring and/or audit purposes."

Example 5 *Cancer stage examination consists of a thorax photo, skeleton scintigraphy and echography of the liver to exclude simultaneous remote metastases.* p. 64, end of Section 3.1

Example 6 *Neo-adjunctive chemotherapy increases the chance for breast saving treatment in locally spread breast cancer and improves the loco-regional control.* p. 67, first conclusion

Example 7 *The prognosis for a patient with locally spread disease is worse than that for a patient with progressed "early disease". Five-year survival rate is 40-60 % and ten-year survival rate is approximately 25 %, depending on the tumor load.* p. 64, Section 3.2

Each guideline in GLIF has its intention described in arbitrary unformatted text. Each statement in GLIF can have a reference to a list of supplemental material. Each item on the list has a purpose and a list of items itself. The purpose can be comment, patient education, tutorial, evidence, or indexing. Each item of supplemental material can contain plain or HTML-formatted text, a drawing, or a movie and has keywords.

In Asbru, each plan has intentions and effects. Intentions contain an expression such as "Bilirubin level is normal" a time annotation describing when this state should occur, and a verb which is one of *avoid*, *maintain*, and *achieve*; *avoid* means it should not happen, *maintain* means it should happen, and *achieve* means it should be achieved.

Effects in Asbru either describe an *argument dependency* or a *plan effect*. The first describes a relationship such as "increased amount of fever reducing drugs leads to a decrease of body

24 Element **background**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
zero or more			
intention	#25, p. 33	zero or more	Intention or goal of the action described
effect	#26, p. 34	zero or more	Expected effect of the action or process
relation	#27, p. 34	zero or more	Expected relation between two entities without known cause
educational	#28, p. 34	zero or more	Information included solely to educate the reader
explanation	#29, p. 34	zero or more	Explanation for related statement
indicator	#30, p. 35	zero or more	Measurement for the success of or compliance to the guideline

This element is used in: *chunk* (#3, p. 19)

25 Element **intention**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
goal	CDATA	required	What to achieve by performing this
modifier	CDATA	optional	Achieve/maintain/avoid for Asbru-style intentions
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *background* (#24, p. 33)

temperature”. *Plan effects* describe the expected change of a parameter (such as fever) during the execution of the plan. In both cases, the likelihood of the effect can be given as percentage.

1.3.6 Resources

Each action consumes resources of various nature:

- *Personal* resources such as the working time of clinical staff.
- *Devices* such as treatment facilities.
- *Financial cost* can be given independent of qualitative or quantitative information on the above items. It includes also drug cost, etc.

Describing personal and device resources might improve the score at AGREE item 19 – “The potential organizational barriers in applying the recommendations have been discussed.”

Describing the financial cost of each task in the guideline relates to AGREE item 20 – “The potential cost implication of applying the recommendations have been considered.” although the AGREE instrument only mentions *considering* them and not explicitly detailing them.

26 Element **effect**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
value-name	CDATA	required	Unique variable name
expected-change	CDATA	required	Calculation rule; or qualitative relation; or global qualitative change
cause	CDATA	optional	Cause of the effect, if known.
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *background* (#24, p. 33)

27 Element **relation**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
left-hand-side	CDATA	required	Right-hand side of relation, e.g., A.
relational-operator	CDATA	required	Operator relating left-hand side to right-hand side, e.g., > or exceeds
right-hand-side	CDATA	required	Right-hand side of relation, e.g., B.
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *background* (#24, p. 33)

28 Element **educational**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
information	CDATA	required	Content of educational information

This element is used in: *background* (#24, p. 33)

29 Element **explanation**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
information	CDATA	required	Content of explanatory information

This element is used in: *background* (#24, p. 33)

30 Element **indicator**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
description	CDATA	required	Description of indicator in any form

This element is used in: *background* (#24, p. 33)

31 Element **resources**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
personal-needed	CDATA	optional	Care personal required
required-devices	CDATA	optional	Medical devices required
financial-cost	CDATA	optional	Financial cost of activity
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *chunk* (#3, p. 19)

1.3.7 Patient related aspects

While the resources dimension mostly represents the view of the care provider, there are several other general issues mentioned in a guideline which see treatment from the patient perspective.

- *Risk* connected to a certain treatment option or diagnostic action.
- *Patient discomfort* is often described as free text and refers to undesired side effects of treatments such as chemotherapy against cancer.
- *Health prospective* can be seen as a resource which is gained by the treatment process, while it is consumed by waiting.

In MHB, each of them is described as free text in a dedicated (optional) attribute of element **patient-aspects**, since the content of these pieces of information tends to be diverse and its integration into a formal model is very complex and as diverse.

Describing these aspects in the guideline may help to improve the score for AGREE item 11: "The health benefits, side effects and risks have been considered in formulating the recommendations."

1.3.8 Document structure

While the position of a statement in the guideline document could be considered a formal property, its status (narrative, displayed recommendation) certainly forms an important context of its interpretation. The various parts are described in Section 1.2.2.

Adding this dimension of annotations to the intermediate representation can help in the automatic formatting of the natural language version of the guideline (which is more a vision than a plan currently) and in the interpretation of the intermediate representation by the creator of the Asbru version of the guideline, who will not have easy access to the natural language text and who might judge a statement differently, depending whether it is found in the narrative text or in a distinguished recommendation.

32 Element **patient-aspects**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
risk	CDATA	optional	Any concern associated with this action (considerable threats with low probability)
discomfort	CDATA	optional	Pain or discomfort for the patient (foreseen but minor with respect to benefits)
health-improvement	CDATA	optional	Expected benefit of the patient from this action
degree-of-certainty	CDATA	optional	Put words like may or sometimes here

This element is used in: *chunk* (#3, p. 19)

33 Element **structure**

<i>Child Name</i>	<i>Element</i>	<i>Occurrence</i>	<i>Comment</i>
gmt-link	#4, p. 19	zero or more	Added by GMT
<i>Attribute Name</i>	<i>Type</i>	<i>Default</i>	<i>Comment</i>
status	introduction	required	Formal status of chunk within guideline
	definition		
	scientific-justification		
	conclusions		
	further-considerations		
	recommendations		
	table		
	other		

This element is used in: *chunk* (#3, p. 19)

There are two settings for the editing of the intermediate representation. First, using the GMT the intermediate representation is seen side by side with the original narrative guideline. Second, using any standard XML editor or other tools, the intermediate representation can be used alone. In this mode of operation, the original text would not be available. To overcome this limitation, we provide slots into which the original text can be copied if desired. This is done by a tool without user interaction. There are two scenarios of usage of the intermediate representation. On the one hand the maintenance of a single living guideline through its various revisions. On the other hand the merging of related guidelines from different organizations.

While copying the original text induces an addition source of inconsistency in the first scenario, it is crucial in the second, in which one will mark up the involved guidelines one by one using the same file of intermediate representation for all of them and then proceed by merging and consolidating the resulting intermediate representation.

1.4 Evaluation

Evaluation of MHB is performed on a theoretical and on a practical level. The practical evaluation is mostly performed during the next phase of the project when the selected guideline will completely be modeled in MHB. This will show any deficits in the representation and potential for future improvement. Hopefully, it will also show that MHB is suitable for modeling clinical guidelines so to build a bridge to its more formal representation in Asbru.

The theoretical evaluation already performed consists of three parts complementing each other:

- The mapping between MHB and various formal representations for clinical guidelines and protocols is discussed in Section 1.3 of this document.
- The suitability of MHB to represent guidelines was discussed with domain experts.
- It was shown that MHB can represent the guideline constructs on a list of patterns currently developed as part of Task 2.5 of this project.

Each of the three activities lead to the conclusion that MHB is sufficiently expressive. The discussion with domain experts also showed that MHB is easier to comprehend than Asbru.

MHB was used to test the Guideline Markup Tool as described in deliverable D2.3a.

Chapter 2

Specification of Asbru Version

This chapter describes Asbru Light, its relation to Full Asbru and the subversions of Asbru Light. In Protocure II, we will not use all the features of Asbru because some of them (e.g., certain parts of the data abstraction) are not necessary to model the guideline we deal with.

The details of the syntax of Asbru are described in the Asbru Reference manual [9]. This chapter is structured similar to that document and refers the reader to that document for details.

The tables found in the following indicate which elements are implemented in which version of Asbru. They use the following symbols:

- fully implemented
- emulated in KIV
- × not part of Asbru Light

Concrete Asbru elements are printed like `this` while the concept behind them is printed in standard English (as opposed to the XML element name) using standard font.

The following sections group the language features into

Plan library discusses the global declarations of the plan library, except for domain definitions and plans which fill sections on their own.

Domain definition specifying the declarative data abstraction part of the library.

Plans describing the attributes of a plan, except its plan body.

Plan body describes the content of the plan, i.e., the steps which are taken on executing this plan.

Basic Elements are used in various places such as data abstraction, conditions and plan body. They are described in the last section together with issues like types, units and data structures.

2.1 Plan Library

The Asbru plan library consists of the following parts:

Library information shows author, date and change history of this plan library.

Domain definitions contain – for one or more different domains – type definitions, context definitions, grouped parameter definitions, value definitions.

Library definitions are similar to domain definitions, but valid for the whole plan library and independent from the domain.

Plans are grouped in plan groups and describe the procedural aspects of the guideline.

2.1.1 Global Declarations

Element `library-info` contains title and version of the library and `administrative-data`, which again contains authors and change history. They are not relevant for verification nor interpretation.

Library definitions contain the same items as domain definitions and are independent from them. Since Asbru Light only supports a single domain per plan library, there is no need for separate library definitions.

2.2 Domain Definition

In Asbru Light, `domain-defs` contains exactly one domain element.

2.2.1 Type Definitions

Records. There are no nested data structures or records in Asbru Light.

Patient record. In Asbru Light, all information about the patient comes via parameters. `patient-record-def` is not used.

Qualitative scale definition.

- primary entries for qualitative scales
- × secondary entries

Numerical scales are important to verify the type of variables and expressions if complex units are used. Simple units are km, h, kg, etc.. Complex units are units composed of more than one simple units, such as km/h.

The first implementation of the Asbru-to-KIV translator only handles simple units and does not interpret `numerical-scale-def`. It is not sure whether the Breast Cancer guideline will require this feature.

Unit definition is not used in Asbru Light.

2.2.2 Parameter Definition

The DTD allows for a rich set of combinations for the abstraction of one parameter based on others. Part of this flexibility is needed to bridge Asbru to the interactive design of data abstraction. Many aspects are useful in high-frequency domains only and therefore not needed in the domain of Breast Cancer.

Children of parameter-def

- × trust-period
- parameter-ref
- raw-data-def
- qualitative-parameter-def
- calculation-def
- logical-combination-def
- × numerical-constant ¹
- × qualitative-constant ¹
- × string-constant ¹
- × constant-ref ¹
- × constant-operation ¹
- × boolean-def
- × change-def
- × logical-dependency-def

The following definitions are useful for high-frequency domains and date-depending domains such as diabetes: average-def, center-def, centile-def, day-of-month, day-of-week, delay-def, duration-def, end-def, end-point-def, episode-analysis-def, episode-def, histogram-def, hour, limit-def, linear-regression-def, minute, month, now, second, selection-def, slope-def, spread-def, standard-deviation-def, standard-error-def, start-def, time-to-alarm-def, time-window-analysis-def, time-window-def, valid-time-def, and year.

Children of calculation-def

- parameter-ref
- × raw-data-def ²
- × qualitative-parameter-def ²
- × string-constant ³
- × logical-combination-def ³
- × qualitative-constant ⁴
- calculation-def
- numerical-constant
- constant-ref
- constant-operation

Children of logical-combination-def

- parameter-ref
- × raw-data-def ²
- × qualitative-parameter-def ²
- × string-constant ³
- logical-combination-def
- × qualitative-constant ⁵
- × calculation-def ³
- comparison-def

¹This would produce a parameter with a constant value.

²Nesting qualitative-parameter-def or raw-data-def as anonymous parameters into calculation-def is not necessary. These parameters should be declared separately (with a name) and referred to using parameter-ref.

³string-constant and logical-combination-def are not type-compatible with the operators provided by calculation-def.

⁴qualitative-constant may be useful together with operators minimum and maximum if ordering of qualitative values is implemented. For the first version of the translator, this is not the case.

⁵If qualitative-constant contains the values true or false, it is not incompatible but redundant, therefore not useful either.

2.2.3 Other Parts of Domain Definition

- global variables
- constants
- context definition

The following elements are not part of Asbru Light:

- Iterator definition is required for lists which are not in Asbru Light.
- Function definition interfaces the plan library to external modules for a certain domain.
- Primitive plan definition interfaces to external code for hardware control etc.
- Various forms of complex time definitions are not needed for applications with simple temporal dimensions such as Breast Cancer.

2.3 Plans

Children of plan

- administrative-data
- explanation⁶
- × derived-from
- refers-to
- × defaults
- arguments
- value-defs
- preferences
- intentions
- conditions
- × state specific pattern
- × effects⁷
- return values

Administrative data, explanation and preferences do not influence the verification and are therefore not translated to KIV.

See Section 2.5.3 for details on the content of intentions and conditions.

2.4 Plan Body

Children of abstract-plan-body

- subplans
- cyclical-plan⁸
- × for-each-plan
- × iterative-plan
- single-step
- refer-to
- × to-be-defined
- user-performed

The support for plan repetition in KIV does not exist currently. However, it is unlikely that there will be forms of repetition in the modeled guideline.

⁶Not relevant for verification, therefore not implemented in KIV.

⁷Effects and background knowledge of the domain will be modelled manually in KIV.

⁸Implemented in KIV to the extend necessary for the modelled guideline.

Features of subplans

- waiting strategy
- retry aborted subplans
- continuation specification
- × propagation specification
- nesting

Children of single-step

- plan-activation
- variable-assignment
- × parameter-assignment
- set-context
- ask
- × list-manipulations
- if-then-else

Both branches in if-then-else contain one or more single-steps.

Features of plan-activation

- on-abort and on-suspend
- arguments and return values
- time-annotation

2.5 Basic Elements

2.5.1 Simple Condition

The simple condition can be part of the conditions of a plan. It is also part of the plan step if-then-else.

Features of simple-condition

- nesting
- comparison
- × range check
- × list checks
- × check whether variable is known

2.5.2 Time Annotation

The standard time annotation consisting of time-range and reference is supported by KIV. But it only supports constants in shifts and for minimum and maximum duration.

Symbolic time annotations

- any
- now
- always
- × time-annotation-ref

Features of reference

- × reference-point (arbitrary expression)
- × epsilon region for reference-point
- plan-state-transition (see below)
- × references
- self
- now

Features of plan-state-transition

- primary plan states
- derived plan states
- direction
- × instance-type
- × instance-number

Features of any-plan-pointer

- simple static-plan-pointer
- × static-plan-pointer specifying invoking plan
- × dynamic plan pointer

2.5.3 Temporal Pattern

Children of temporal-pattern

- constraint-combination
- constraint-not
- × count-constraint
- none
- parameter-proposition
- parameter-ref
- plan-state-constraint
- refer-to
- simple-condition
- × temporal-constraint
- × timeless-parameter-query
- × to-be-defined

Children of parameter-proposition

- value-description
- value-range
- is-known-parameter
- context
- time-annotation
- × sampling-frequency
- × importance

2.5.4 Expression

Children of expression

- argument-ref
- constant-ref
- × data-ref
- × function-call
- × get-position
- now
- numerical-constant
- operation
- parameter-ref
- plan-state-transition
- qualitative-constant
- × string-constant

Chapter 3

Specification of Asbru in KIV

3.1 Introduction

The goal of this section is to define a representation of Asbru in KIV. KIV is an interactive theorem prover which supports algebraic specifications and higher order logic with special support for temporal logic. Temporal logic is used to express properties of concurrent systems. Concurrent systems can be defined as parallel programs or, alternatively, as state machines. Protocure I has shown that Asbru plans also define concurrent systems. In Protocure I, we have manually translated Asbru plans to (enriched) parallel programs. Our goal for Protocure II is to more directly support Asbru in KIV: we will define a representation which is as close as possible to the Asbru language and expand the calculus to allow for its symbolic execution. The language is described here, and symbolic execution is part of Deliverable 4.2b.

Figure 3.1 contains the algebraic specification of plan OBSERVATION which is part of the jaundice case study of Protocure I [1]. The specification is an example of an Asbru plan which has been manually translated to a parallel program. The program already utilises a special construct **break α if φ** which is similar to an interrupt. The construct has been defined to directly support conditions of Asbru. Other features of Asbru have not been directly supported but were encoded which lead to a translation overhead. For example, the **wait for** construct of the parent plan REGULAR-TREATMENTS required an additional variable *OB-State* to explicitly store the current state of plan OBSERVATION. As a result, the translated plan contains three additional assignments. Individual solutions for several features of Asbru have been used. Furthermore, only for those features of Asbru which were used in the two case studies of Protocure I, translation patterns to KIV have been defined.

Instead of translating Asbru plans to parallel programs, we will here define a representation in KIV which is as close as possible to Asbru; especially the translation overhead should be avoided and all features should be directly supported. Furthermore, we will consider a number of features which were not yet supported: setup precondition, suspend and resume conditions, and time annotations.

An overview of our solution is depicted in Figure 3.2. We will define a representation of Asbru in KIV as an algebraic specification in Section 3.3. Beforehand, we define in Section 3.2 an XML representation of the specification which is used to link the KIV representation to Asbru. The XML representation is even closer to Asbru than the algebraic specification. We use a simple XSL transformation to transform the XML facade to the underlying algebraic specification (see Section 3.4).

3.2 XML Input Format for KIV

As part of the task, to make the translation of an Asbru plan to a KIV specification less error-prone, we have defined a new interface to the KIV system. This interface is defined in the XML.

```

enrich Asbru-basic with

predicates
  ob-filter : patient-data;
  ob-abort : patient-data;

procedures
  observation# : patient-data  $\times$  nat  $\times$  plan-state;
  prescribe-observation# : nat;

variables
  OB-State: plan-state flexible;
  ob-state: plan-state;

axioms
  filter-def :
  ob-filter(patient-data)  $\leftrightarrow$  get-bilirubin(patient-data.tsb) = observation;
  abort-def :
  ob-abort(patient-data)  $\leftrightarrow$  get-bilirubin(patient-data.tsb)  $\neq$  observation;

declaration
  observation#(Patient-Data, Time, OB-State)
    begin
      await ob-filter(Patient-Data);
      OB-State := activated;
      pbreak prescribe-observation#(; Time) if ob-abort(Patient-Data);
      if ob-abort(Patient-Data)
      then OB-State := aborted
      else OB-State := completed
    end;

  prescribe-observation#(Time)
    begin
      skip; sleep#(; Time)
    end;

end enrich

```

Figure 3.1: Asbru plan OBSERVATION manually translated to parallel programs

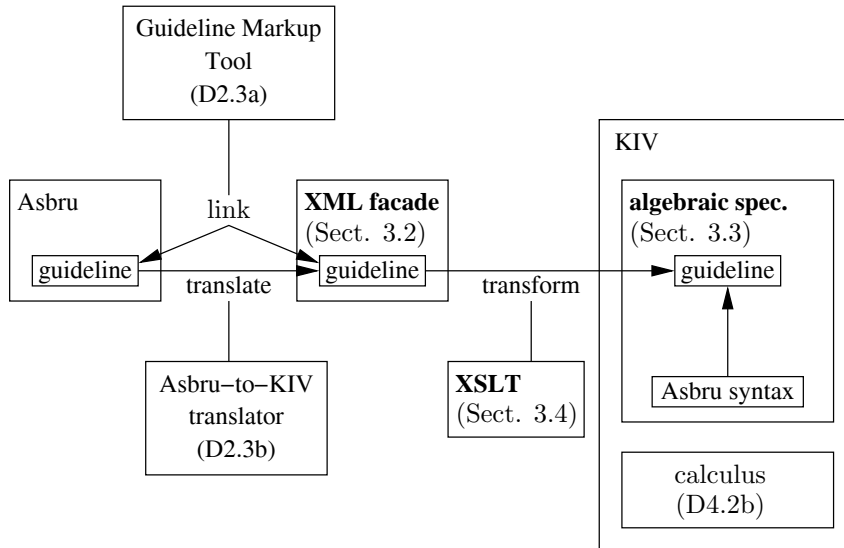


Figure 3.2: Overview of KIV representation

Using XML allows a user to perform basic validity checks even without access to the KIV parser. In conjunction with the much more fine coarsed error messages, XML parsers tend to provide, it enables the user to identify the malformed input faster.

Additionally, the use of XML makes it possible to link the KIV specification parts to the corresponding Asbru parts using the Guideline Markup Tool (GMT). This will provide a base for the upcoming task of the formal verification of selected guidelines, especially tracing back the verification results into Asbru: once the source of error is identified within the KIV specification, the GMT will be able to automatically track it back into the Asbru language, from there to the intermediate representation and finally into the informal guideline.

Appendix 3.6 contains the data type definition (DTD) of the KIV representation of Asbru. Here, we present the KIV representation of the Asbru plan OBSERVATION in Figure 3.3. For this example, we will demonstrate the ease of translation and compare the XML input to the Protocure I style input. A reader, familiar with the Asbru language should recognize the relevant parts and see, that a translation from Asbru into the new XML format is pretty much straightforward.

Every Asbru plan is translated into an enrichment, where the plan defines an axiom. The element

```
<asbru-plan planname="Observation" lemmaname="ob-def">
...
</asbru-plan>
```

is the wrapper declaring begin and end of the Asbru plan. The name of the plan and the corresponding axiom name is provided here.

```
<control type="sequential">
  <subplan>Prescribe Observation</subplan>
</control>
```

This section describes the control, that is running concurrently to the plan. The semantics of these controls is defined by [2]. Currently all controls are implemented. The subplan sections provide the identifiers of all potentially running direct subplans. These identifiers are unique.

```
<conditions>
...
</conditions>
```

```

<enrich>
  <axioms>
    <asbru-plan planname="Observation" lemmaname="ob-def">
      <control type="sequential">
        <subplan>Prescribe Observation</subplan>
      </control>
      <conditions>
        <filter-precondition overridable="false"
                           confirmation_required="false">
          <simple-constraint>
            lambda patient-data-history, variable-history, asbru-state-history,
                    asbru-state, asbru-clock.
              get-bilirubin(patient-data-history[asbru-clock].tsb)
                = observation
          </simple-constraint>
        </filter-precondition>
        <abort-condition overridable="false"
                       confirmation_required="false">
          <simple-constraint>
            lambda patient-data-history, variable-history, asbru-state-history,
                    asbru-state, asbru-clock.
              not get-bilirubin(patient-data-history[asbru-clock].tsb)
                = observation
          </simple-constraint>
        </abort-condition>
      </conditions>
      <intentions>
        <intention type="intermediate-state" verb="maintain">
          <parameter-proposition>
            <time-annotation>
              <ess>hour(4)</ess>
              <lfs>hour(6)</lfs>
              <referencePoint>*self*</referencePoint>
            </time-annotation>
            <simple-constraint>
              lambda pdh, vh, ash, as, ac. pdh[ac].tsb < pdh[ash.acttime]
            </simple-constraint>
          </parameter-proposition>
        </intention>
      </intentions>
    </asbru-plan>
  </axioms>
</enrich>

```

Figure 3.3: KIV representation of plan OBSERVATION

Within this section, the conditions, that control the execution of the plan are defined. As each plan definition is done according to the same pattern, we present this detailed definition only for the filter precondition. Other conditions are defined the same way.

```

<filter-precondition overridable="false"
                    confirmation_required="false">
  <simple-constraint>
    lambda patient-data-history, variable-history, asbru-state-history,
          asbru-state, asbru-clock.
          get-bilirubin(patient-data-history[asbru-clock].tsb)
          = observation
  </simple-constraint>
</filter-precondition>

```

The filter precondition of OBSERVATION is rather simple. It has no time annotation and is therefore only a `simple-constraint`.

A more complex type of such a condition is the `parameter-proposition`, which includes a time-annotation. Within the time-annotation section, the temporal bounds of the condition are defined. We expect any given time-annotation to be legal, which is, the starting point of any interval is earlier than the corresponding finishing point (e.g. `ess` is earlier than `lss`). Also, there are more subtle requirements. For example, the `ess` plus the `minDuration` is less than or equal the `lfs`. This is a necessary criterion that there is a possibility to fulfill the condition. For more details on time annotations, see [8]. A time annotation is written in XML as

```

<time-annotation>
  <ess>ess_time</ess>
  <lss>lss_time</lss>
  <efs>efs_time</efs>
  <lfs>lfs_time</lfs>
  <minDuration>min_time</minDuration>
  <maxDuration>max_time</maxDuration>
  <referencePoint>ref_time</referencePoint>
</time-annotation>

```

Additionally, conditions can consist of logical combinations of parameter propositions. Allowed combinations are `and`, `or`, `xor` and `not`.

These are the rest of the conditions that can be defined.

```

<setup-precondition ...>
  ...
</setup-precondition>
<suspend-condition ...>
  ...
</suspend-condition>
<reactivate-condition ...>
  ...
</reactivate-condition>
<complete-condition ...>
  ...
</complete-condition>
<abort-condition ...>
  ...
</abort-condition>

```

Any of these conditions omitted are assigned default values, that do not prevent the plan from running.

For technical reasons, intentions are defined separately from the Asbru plan. The element

```

<intentions>
  <intention>
    ...
  </intention>
</intentions>

```

defines an intention, where the dotted part can be filled with an arbitrary (time annotated) condition.

Compared to the Procure I style translation of plan OBSERVATION in Figure 3.1, it can easily be seen, that the translation from Asbru into the old KIV format is much more complicated and error-prone. Therefore, we consider the KIV representation in XML an important step towards the every day application of the proposed formal methods.

Note: Parts of the XML input format might be subject to change, once first real world tests are conducted and show the demand for a change in format.

Note: It is **strongly** suggested, not to use the KIV ASCII input format for asbru plans any more!

3.3 Algebraic Specifications

3.3.1 Example

Figure 3.4 contains the algebraic specification of plan OBSERVATION which is the result of the transformation of the XML facade. An Asbru plan is defined using an algebraic function `mk-asbru-def`, the parameters being as follows.

- The first six parameters are filled with the different conditions. These are (in order) the filter-precondition, setup-precondition, suspend condition, reactivate condition, complete condition and abort condition. Conditions, that are not specified in the XML facade, are translated into default values, where such a default value does not prevent the plan from running. For example, a filter-precondition will always evaluate true, if not specified, an abort condition will always evaluate to false. The conditions are of type `asbru-condition`.
- The next parameter determines the control type of the subplans of this plan. These can be `sequential`, `parallel`, `unordered`, `anyorder`, `if-then-else`, `if-then`, `cyclical`, `onsuspend`, `onabortonsuspend` or `onabort`. In addition it can be `user` to indicate that the plan is user-performed.
- Next parameter is the retry value. It specifies the behavior of the children of this plan, once they have been aborted. If `retry` is set, the parent will send its children the retry signal, thus trying to reexecute the plan. Otherwise, the planstate remains at `aborted`.
- The subplans are specified by the next parameter. It is a list containing all subplans of this superplan. If a plan has no subplans, the empty list is used. It is denoted as `@`.
- Next in the specification is the wait-for section. `wait-for` is a data type, containing the subplans, the superplan has to wait for unless it can set its state to completed. These plans can be composed by the usual logic operators `and`, `or`, `xor` or `not`.
- Last in the specification is the optional-waitfor flag. An Asbru plan might spawn more subplans, than it actually does need to complete (i.e. satisfy complete condition and wait-for statements). There are two options how to proceed, once a plan is ready to complete but some of the subplans are still running. Those subplans can either be aborted or the superplan can wait for their completion. The latter is done, if the optional-waitfor flag is set, the first, otherwise.

enrich Asbru with

axioms

```
ob-def :
asbru('ob') = mk-asbru-def(
  mk-asbruc(
    mk-ccond( $\lambda$  pdh, vh, ash, as, ac. pdh[ac]['bilirubin'] .val = observation),
    false, false
  ),
  setup_condition,
  suspend_condition,
  resume_condition,
  complete_condition,
  mk-asbruc(
    mk-ccond( $\lambda$  pdh, vh, ash, as, ac. pdh[ac]['bilirubin'] .val  $\neq$  observation),
    false, false
  ),
  sequential,
  false,
  'pob' ,
  bwf(pob),
  false
);

intention-nr : # intentions('ob') = 1;

intention-1 :
intentions('ob')[0] = mkintention(
  intermediate-state,
  maintain,
  mk-ta-ccond(
    mk-ccond( $\lambda$  pdh, vh, ash, as, ac. pdh[ac].tsb < pdh[ash.acttime])
    mk-cta(
      hour(4), unspecified_max_time,
      unspecified_min_time, hour(6),
      mk-asbru-clock(0), unspecified_max_time,
      *self*
    )
  )
);
```

end enrich

Figure 3.4: KIV specification of OBSERVATION

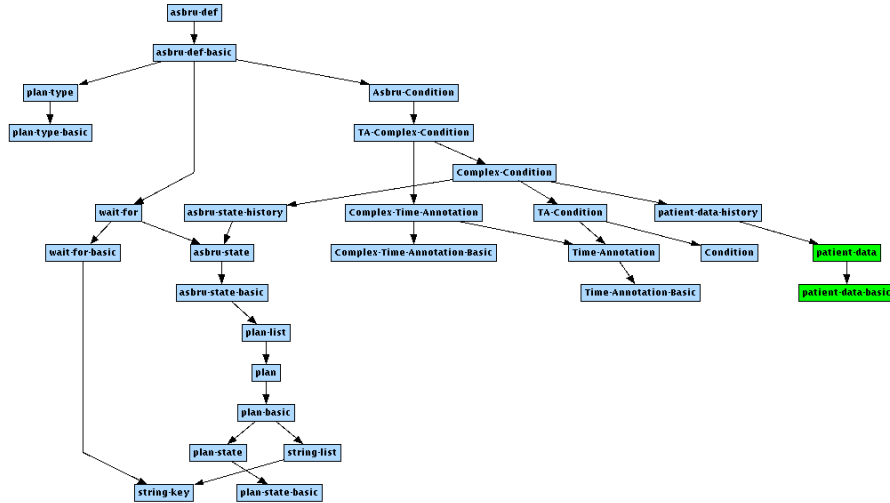


Figure 3.5: Definition of syntax of Asbru plans

The algebraic function intentions maps the plan key ‘ob’ to its list of intentions. A first axiom defines the number of intentions – the hash function # returning the length of a list. Further axioms define the intentions. An intention is constructed with a function mkintention which takes three parameters:

- the first parameter defines the type of the intention,
- the second parameter defines the so called verb of the intention, and
- the third parameter contains the intended condition.

3.3.2 Syntax specification

The syntax of Asbru is defined with several algebraic specifications in KIV. Figure 3.5 gives an overview of the specifications and especially illustrates the dependency structure. The essential part of the specification is an algebraic type “asbru-def” in KIV. An “asbru-def” simply defines a structure of the following form:

```

asbru-def
= mk-asbru-def
(. .filter : asbru-condition;
 . .setup : asbru-condition;
 . .suspend : asbru-condition;
 . .reactivate : asbru-condition;
 . .complete : asbru-condition;
 . .abort : asbru-condition;
 . .type : plan-type;
 . .retry : bool;
 . .subplans : string-list;
 . .waitfor : wait-for;
 . .opt-wf : bool;
);

```

The conditions are of type **asbru-condition** which is defined in the corresponding specification ASBRU-CONDITION. They can be selected out of the asbru-def construct by the corresponding selectors (e.g. .filter for the filter-precondition).

The plan type describes the way of execution of the subplans. Some of the possible values, this slot can take, are sequential, parallel or user.

With the retry flag set, the plan is allowed to restart its children, once they are aborted. Predicates have been implemented to determine, whether a plan is still fulfillable or not.

The subplans slot contains plan keys of sub plans, that can be started as children of this plan. The way, they are to be started is determined by the plan type. It is possible for a plan to complete, without starting all subplans.

In KIV there are four types of conditions specified to match the expressiveness of Asbru and at the same time provide means for easier evaluation of the truth value of the conditions. In principle, a condition a **patient-data-history**, a **variable-history**, an **asbru-state-history**, an **asbru-state** and an **asbru-clock** to a boolean.

The differences concern the asbru clock, which may be a basic asbru-clock, denoting an absolut timepoint or a more complex type of asbru clocks, that allow also for timepoints relative to plan state changes and the current time. Asbru specifies, that each condition is accompanied by two booleans, that denote the influence of the user on the evaluation of the condition. It can be specified, that the user is allowed to override the condition or to prevent it to become true, if he sees fit.

The most complex condition defined in KIV is the **abstract-absru-condition** which incorporates the complex time points as well as the booleans.

Along with these conditions, **time annotations** are defined. Those are aggregations of seven asbru-clocks. For details on time annotations please refer to [8].

3.4 XSL Transformation

The algebraic specification of KIV and the XML input format closely resemble each other. The content of each condition tag in the XML specification is transformed into the corresponding condition slot within mk-asbru-def.

This transformation is pretty much straight forward. Time annotations are translated into a mk-time-annotation construct, where the time slots are filled from the similar named xml slots. Undefined slots are filled with default values. Default value for lss, lfs and maxDuration is positive infinity, for ess and efs negative infinity and for minDuration zero. This work is fully automated by XSL transformations. Therefore this work is done on a syntactical level.

Each simple-constraint has to be given in form of a function, mapping one patient-data-history, one variable-history, one asbru-state-history, one asbru-state and one asbru-clock (in that specific order) to a boolean. In KIV (and the underlying PPL) such functions can be denoted as anonymous lambda functions. An example for such a function is

$$\lambda \text{ ta, c, ac. satisfied}(c, \text{ta}, \text{ac})$$

λ denotes the beginning of such an anonymous function. Separated by commas are the input parameters. A dot after the last parameter is the key symbol for the beginning of the function itself. In case of the above definition, the result of the function is the return value of the predicate **satisfied**. Such predicates have to be defined either in the corresponding or the parent specification.

Each simple-constraint is then translated into a mk-acond construct. The resulttype of this constructor is a asbru-condition. A constraint combination combines two asbru-conditions into one new asbru-condition. The combination can be done by an `_or` or an `_and`. These operators map down to first order logic, i.e. an and-complex-condition is true in any state where both parts of the combination are true.

3.5 Open Issues

A number of features of Asbru have not been considered for the KIV representation; the more important missing feature is written down below. We will add support for these features, if they

arise in the breast cancer case study.

Multiple plan instance

At the moment the string key identifies the plan in `asbru-state` and in `asbru-def`. If the same plan can run in different instances, we have to expand the key in `asbru-state` by an ID. The string key will still identify which plan it is and the ID will distinguish between different instances.

3.5.1 Changes to previous version of this paper

A few items were changed during revision of this deliverable. In contrast to the 2004 version of this paper, cyclical plans are now supported within Asbru KIV. Due to the complexity of cyclical plans it has been decided to implement the subset of cyclical plans, necessary for the verification of breast cancer. The semantics definition has been changed accordingly.

The if-then-else construct is now fully supported within KIV and has been accompanied by a if-then plan control without the else branch.

Minor changes of the syntax of conditions have been made to match the syntax of this document and the actualized Asbru KIV syntax.

3.6 DTD for Asbru plans in KIV

This is the dtd for the current XML KIV input format. We designate this dtd as version 0.9 as we expect changes, once it is used for real world application.

```
<!ELEMENT spec1 (basicspec | unionspec | genericspec | enrichedspec |
    instantiatedspec | actualizedspec | renamedspec |
    basicdataspec | gendataspec | chartspec | asmspec)>

<!ELEMENT basicspec (specpart, comment?)>
<!ELEMENT unionspec (usedname+, comment?)>
<!ELEMENT genericspec (paramname+, usedname*, specpart, comment?)>
<!ELEMENT enrichedspec (usedname+, specpart, comment?)>
<!ELEMENT instantiatedspec (paramname*, usedname+, symmaplist,
    extsymrenlist, comment?)>
<!ELEMENT actualizedspec (genname+, instname+, symrenlist, comment?)>
<!ELEMENT renamedspec (usedname+, symrenlist, comment?)>
<!ELEMENT gendataspec (paramname+, usedname*, datasortdef+,
    vardef+, fctdef*, prddef*, comment?)>
<!ELEMENT chartspec (usedname*, chart+, comment?)>
<!ELEMENT basicdataspec (usedname*, datasortdef+, vardef+,
    fctdef*, prddef*, comment?)>
<!ELEMENT asmspec (procsym, usedname*, asmpart?, comment?)>

<!ELEMENT chart (orchart | andchart | basicchart)>
<!ELEMENT orchart (orchart_basic+, chartsig, init+,
    trans+, staticreaction+, comment?)>
<!ELEMENT andchart (andchart_basic+, chartsig, comment?)>
<!ELEMENT basicchart (vardecl, chartsig, staticreaction+, comment?)>
<!ELEMENT chartsig (vardef+, evsym+, vardef+, evsym+)>
<!ELEMENT init (vardecl+)>

<!ELEMENT specpart (signature,gen*,(lemma|asbru-plan)*,decl*)>
<!ELEMENT usedname (#PCDATA)>
<!ELEMENT paramname (#PCDATA)>
```

```

<!ELEMENT instname (#PCDATA)>
<!ELEMENT genname (#PCDATA)>

<!ELEMENT asmpart (signature,
                    (inputvar*, statevar*, prog,
                     expr, procsym, predecl+)?)>
<!ELEMENT predecl (preprog_nonppl)>
<!ELEMENT prog (prog_nonppl | ppl_variable)>

<!ELEMENT symmaplist (sortmap | opvarprocmap)*>
<!ELEMENT extsymrenlist (sortren | extfctren | procren | extvarren)*>
<!ELEMENT symrenlist (sortren | fctren | fctren1 | procren | varren)*>
<!ELEMENT signature (sortdef*, constdef*, fctdef*,
                     prddef*, procdef*, vardef*)>

<!ELEMENT gen (sort+, extfct+, comment?)>
<!ATTLIST gen name NMTOKEN #IMPLIED>
<!ATTLIST gen type NMTOKEN #IMPLIED>

<!ELEMENT sort (#PCDATA)>
<!ELEMENT inputvar (var)>
<!ELEMENT statevar (var)>

<!ELEMENT asbru-plan (control, conditions, intentions)>
<!ATTLIST asbru-plan planname NMTOKEN #REQUIRED>
<!ATTLIST asbru-plan lemmaname NMTOKEN #REQUIRED>

<!ELEMENT control (retry?, subplan*, wait-for?, optional-waitfor?)>
<!ATTLIST control type (sequential|unordered|parallel|anyorder|
                       onabort|ifthen|ifthenelse|user|cyclical) #REQUIRED>

<!ELEMENT cyclical-plan (cyclical-time-annotation, times-completed?)>

<!ELEMENT cyclical-time-annotation (ess?, lss?, efs?, lfs?, minDuration?,
                                     maxDuration?, referencePoint, offset, frequency)>
<!ELEMENT times-completed (#PCDATA)>
<!ELEMENT frequency (#PCDATA)>

<!ELEMENT waitfor (simple-waitfor|complex-waitfor|negated-waitfor)*>

<!ELEMENT simple-waitfor (#PCDATA)>
<!ELEMENT negated-waitfor (waitfor)>
<!ELEMENT complex-waitfor (waitfor, waitfor)>
<!ATTLIST complex-waitfor type (and|or|xor|not)>

<!ELEMENT retry EMPTY>
<!ELEMENT optional-waitfor EMPTY>

<!ELEMENT subplan (#PCDATA)>

<!ELEMENT intentions (intention*)>

<!ELEMENT intention (constraint-not|constraint-combination|
                    simple-constraint|parameter-proposition)>

```

```

<!ATTLIST intention type (intermediate-state|intermediate-action|
                           overall-state|overall-action)>
<!ATTLIST intention verb (avoid|maintain|achieve)>

<!ELEMENT conditions (filter-precondition?, setup-precondition?,
                      suspend-condition?, reactivate-condition?,
                      complete-condition?, abort-condition?)>

<!ELEMENT filter-precondition (constraint-not|constraint-combination|
                               simple-constraint|parameter-proposition)+ >
<!ATTLIST filter-precondition overridable NMTOKEN #REQUIRED>
<!ATTLIST filter-precondition confirmation_required NMTOKEN #REQUIRED>

<!ELEMENT setup-precondition (constraint-not|constraint-combination|
                              simple-constraint|parameter-proposition)+>
<!ATTLIST setup-precondition overridable NMTOKEN #REQUIRED>
<!ATTLIST setup-precondition confirmation_required NMTOKEN #REQUIRED>

<!ELEMENT suspend-condition (constraint-not|constraint-combination|
                             simple-constraint|parameter-proposition)+>
<!ATTLIST suspend-condition overridable NMTOKEN #REQUIRED>
<!ATTLIST suspend-condition confirmation_required NMTOKEN #REQUIRED>

<!ELEMENT reactivate-condition (constraint-not|constraint-combination|
                                simple-constraint|parameter-proposition)+>
<!ATTLIST reactivate-condition overridable NMTOKEN #REQUIRED>
<!ATTLIST reactivate-condition confirmation_required NMTOKEN #REQUIRED>

<!ELEMENT complete-condition (constraint-not|constraint-combination|
                              simple-constraint|parameter-proposition)+>
<!ATTLIST complete-condition overridable NMTOKEN #REQUIRED>
<!ATTLIST complete-condition confirmation_required NMTOKEN #REQUIRED>

<!ELEMENT abort-condition (constraint-not|constraint-combination|
                           simple-constraint|parameter-proposition)+>
<!ATTLIST abort-condition overridable NMTOKEN #REQUIRED>
<!ATTLIST abort-condition confirmation_required NMTOKEN #REQUIRED>

<!ELEMENT constraint-not (constraint-not|constraint-combination|
                          simple-constraint|parameter-proposition)+>

<!ELEMENT constraint-combination (constraint-not|constraint-combination|
                                  simple-constraint|parameter-proposition)+>
<!ATTLIST constraint-combination type (and|or|xor) #REQUIRED>

<!ELEMENT parameter-proposition (time-annotation, simple-constraint)>

<!ELEMENT simple-constraint (#PCDATA)>

<!ELEMENT time-annotation (ess?, lss?, efs?, lfs?, minDuration?,
                           maxDuration?, referencePoint)>
<!ELEMENT ess (#PCDATA)>
<!ELEMENT lss (#PCDATA)>

```

```
<!ELEMENT efs (#PCDATA)>
<!ELEMENT lfs (#PCDATA)>
<!ELEMENT minDuration (#PCDATA)>
<!ELEMENT maxDuration (#PCDATA)>
<!ELEMENT referencePoint (#PCDATA)>

<!ELEMENT comment (#PCDATA)>
<!ELEMENT lemma (#PCDATA)>
<!ELEMENT var (#PCDATA)>
<!ELEMENT decl (#PCDATA)>
<!ELEMENT expr (#PCDATA)>
<!ELEMENT vardecl (#PCDATA)>
<!ELEMENT extfct (#PCDATA)>
<!ELEMENT sortdef (#PCDATA)>
<!ELEMENT constdef (#PCDATA)>
<!ELEMENT fctdef (#PCDATA)>
<!ELEMENT prddef (#PCDATA)>
<!ELEMENT procdef (#PCDATA)>
<!ELEMENT vardef (#PCDATA)>
<!ELEMENT fctren (#PCDATA)>
<!ELEMENT fctren1 (#PCDATA)>
<!ELEMENT ppl_variable (#PCDATA)>
<!ELEMENT sortmap (#PCDATA)>
<!ELEMENT opvarprocmap (#PCDATA)>
<!ELEMENT evsym (#PCDATA)>
<!ELEMENT sortren (#PCDATA)>
<!ELEMENT extfctren (#PCDATA)>
<!ELEMENT procren (#PCDATA)>
<!ELEMENT extvarren (#PCDATA)>
<!ELEMENT varren (#PCDATA)>
<!ELEMENT datasortdef (#PCDATA)>
<!ELEMENT orchart_basic (#PCDATA)>
<!ELEMENT andchart_basic (#PCDATA)>
<!ELEMENT trans (#PCDATA)>
<!ELEMENT staticreaction (#PCDATA)>
<!ELEMENT procsym (#PCDATA)>
<!ELEMENT preprog_nonppl (#PCDATA)>
<!ELEMENT prog_nonppl (#PCDATA)>
```

Bibliography

- [1] Deliverable 3 of EU project Protocure I, 2002. available at www.protocure.org.
- [2] M. Balsler, C. Duelli, and W. Reif. Formal semantics of Asbru – an overview. In *Proceedings of IDPT 2002*. Society for Design and Process Science, 2002.
- [3] The AGREE Collaboration. Appraisal of guidelines for research and evaluation (AGREE) instrument, 2001. ISBN 1 8981 8321 X.
- [4] M. J. Field and K. H. Lohr. *Guidelines for Clinical Practice. From Development to Use*. National Academy Press, 1992.
- [5] J. Fox, N. Johns, C. Lyons, A. Rahmzadeh, R. Thomson, and P. Wilson. Proforma: A general technology for clinical decision support systems. *Computer Methods and Programs in Biomedicine*, 54:59–67, 1997.
- [6] Nationaal Borstkanker Overleg Nederland (NABON). *Guideline for the treatment of breast carcinoma*. Van Zuiden Communications B.V., 2002.
- [7] M. Peleg, A. Boxwala, S. Tu, R. Greenes, E. Shortliffe, and V. Patel. Handling expressiveness and comprehensibility requirements in GLIF3. In *Proceedings of the 10th World Congress on Medical Informatics (MedInfo 2001)*, pages 241–245, London, 2001.
- [8] A. Seyfang. *An Integrated System for Temporal Data Abstraction to Facilitate Guideline Execution and Knowledge-Based Data Analysis*. PhD Thesis, Vienna University of Technology, 2005, to appear.
- [9] A. Seyfang, R. Kosara, and S. Miksch. Asbru 7.3 reference manual. Technical report, Vienna University of Technology, 2002.
- [10] Y. Shahar, O. Young, E. Shalom, A. Mayaffit, R. Moskovitch, A. Hessing, and M. Galperin. DEGEL: A hybrid, multiple-ontology framework for specification and retrieval of clinical guidelines. In *Proceedings in Artificial Intelligence in Medicine*, 2003.
- [11] R. N. Shiffman, B. T. Karras, A. Agrawal, R. Chen, L. Marengo, and S. Math. Gem: A proposal for a more comprehensive guideline document model using xml. *Journal of the American Medical Informatics Association*, 7(5):488–498, 2000.
- [12] V. Svatek and M. Ruzicka. Step-by-step mark-up of medical guideline documents. *International Journal of Medical Informatics*, 70(2-3):329–335, 2003.
- [13] P. Terenziani, G. Molino, and M. Torchio. A modular approach for representing and execution clinical guidelines. *Artificial Intelligence in Medicine*, 23:249–276, 2001.