

From Informal Knowledge to Formal Logic: A Realistic Case Study in Medical Protocols

Mar Marcos^{1*}, Michael Balsler², Annette ten Teije³, and Frank van Harmelen¹

¹ Vrije Universiteit Amsterdam, Dept. of Artificial Intelligence
De Boelelaan 1081a, 1081HV Amsterdam, Netherlands

² Universität Augsburg, Lehrstuhl Softwaretechnik und Programmiersprachen
86135 Augsburg, Germany

³ Universiteit Utrecht, Institute of Information and Computing Sciences
P.O. Box 80.089, 3508TB Utrecht, Netherlands

Abstract. We report our experience in a case study with constructing fully formalised knowledge models of realistic, specialised medical knowledge. We have taken a medical protocol in daily use by medical specialists, modelled this knowledge in a specific-purpose knowledge representation language, and finally formalised this knowledge representation in terms of temporal logic and parallel programs. The value of this formalisation process is that each successive formalisation step has contributed to improving the quality of the original medical protocol, and that the final formalisation allows us to provide machine-assisted proofs of properties that are satisfied by the original medical protocol (or, alternatively, precise arguments why the original protocol fails to satisfy certain desirable properties). We believe that this is the first time that a significant body of medical knowledge (in our case: a protocol for the management of jaundice in newborns) has been formalised to the extent that it becomes amenable to automated theorem proving, and that this has actually lead to improvement of the original body of medical knowledge.

1 Introduction

During the last years a high number of medical practice guidelines or protocols¹ have been produced from systematic evidence-based reviews [1]. They are “systematically developed statements to assist practitioners and patient decisions about appropriate health care for specific circumstances” [2]. Medical protocols contain more or less precise recommendations about the diagnosis tests or the interventions to perform, or about other aspects of clinical practice. These recommendations are based on the best empirical evidence available at the moment. Among the potential benefits of protocols, we can highlight the improvement of healthcare outcomes [3]. More precisely, they can help to promote high quality practice, recommending interventions of proved benefit and discouraging those that are not supported by good evidence. They can also be used

* On research leave from the Dept. of Computer Engineering and Science, Universitat Jaume I, Castellón, Spain.

¹ In this paper we use the terms guideline and protocol indistinctively. However, the term protocol is in general used for more specific versions of a guideline.

to reduce variations in care. Finally, protocols can be useful to improve cost efficiency, thanks to the standardisation of healthcare. Indeed, it has been shown that adherence to protocols may reduce the costs of care upto 25% [4].

In order to enable their potential benefits, protocols must fulfill strong quality requirements. This is true not only for the final product, the protocol, but also for the development process. Medical bodies worldwide have made efforts in this direction, e.g. elaborating appraisal documents that take into account a variety of protocol aspects, of both protocols and their development process (see [5] for a comparison of appraisal instruments). However, these initiatives are not sufficient since they rely on informal methods and notations. We are concerned with a different approach, namely the quality improvement of medical protocols through formalisation. Currently, protocols are described using a combination of different formats, e.g. text, flow diagrams and tables. The underlying idea of our work is that making these descriptions more precise, with the help of a more formal language, will expose parts where the protocols are ambiguous, incomplete or even inconsistent. By pointing out these anomalous parts, and the reasons why they could be problematic, we expect to obtain useful indications for the improvement of the protocols. This idea is widely acknowledged in fields like software engineering, where formal methods are used as a tool for early detection of specification and design errors, but has been largely unexplored for medical protocols.

However, the formalisation of medical protocols can be tackled at different degrees of formality. In this paper we aim at a fully formal specification. The research question that we try to answer is: *can medical protocols be formalised in terms of logic? and can this formalisation contribute to the improvement of their quality?* In order to answer this question, we have carried out a case study on protocol formalisation. The main contribution of this paper is to show (1) that it is possible to formalise medical knowledge to a high degree of formality; (2) that this process must be divided into a number of steps, each increasing the degree of formality; and (3) that each step in this process uncovers problems in the protocol. An early report on similar problems in protocols can be found in [6]. It is important to notice that our work differs in significant aspects, namely that we aim at a much higher degree of formality, and that we focus on the verification and validation of the original protocol rather than the design of an enhanced version thereof.

For the purpose of our case study, a choice had to be made on the target formalism(s) as well as on the medical protocol to be used. Concerning the formalisms, we have first used a special purpose knowledge representation language suited for medical protocols—Asbru, and afterwards the logic of the KIV theorem prover. As for the medical protocol, we have selected one devoted to the the management of jaundice in newborn babies. Figure 1 illustrates the process of our case study, and also the structure of this paper. First the jaundice protocol is discussed in section 2. Then the Asbru language and the model of the jaundice protocol in this language are described in section 3. The next step in the formalisation process is to translate the Asbru protocol to the fully formal calculus of KIV. This step is described in section 4. In each of the previous two sections we also discuss the benefits of the corresponding formalisation step, as well as the difficulties we encountered. Finally, section 5 concludes the paper.

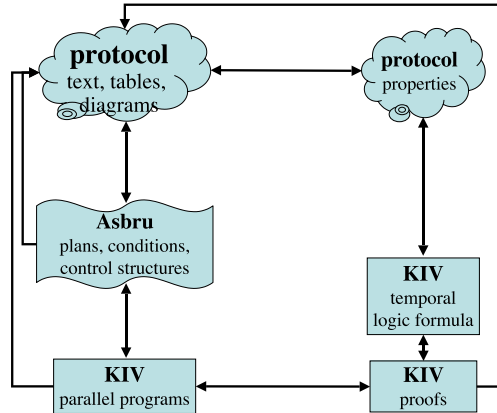


Fig. 1. The formalisation process of our case study.

2 The Jaundice Protocol

Jaundice (or hyperbilirubinemia) is a common disease in newborn babies. Under certain circumstances, elevated bilirubin levels may have detrimental neurological effects. In many cases jaundice disappears without treatment but sometimes phototherapy is needed to reduce the levels of total serum bilirubin (TSB), which indicates the presence and severity of jaundice. In a few cases, however, jaundice is a sign of a severe disease.

The jaundice protocol of the American Association of Pediatrics² (AAP) [7] is intended for the management of the disease in healthy term³ newborn babies. The main reason for choosing this protocol was that it is considered a high-quality protocol: the jaundice protocol of the AAP is included in the repository of the National Guideline Clearinghouse⁴.

The guideline is a 10 pages document which contains knowledge in various informal forms, namely:

- text (this is the main body of the protocol),
- a list of factors to be considered when assessing a jaundice infant (for instance, family history of significant hemolytic disease),
- two tables, one for the management of Hyperbilirubinemia in the healthy term newborn and another for the treatment options for jaundice breast-fed infants, and
- a flowchart-like notation representing the steps described in the guideline.

The protocol consists of an evaluation (or diagnosis) part and a treatment part, to be performed in sequence. During the application of the protocol, as soon as the possibility of a more serious disease is uncovered, the recommendation is to exit without any further action. The rationale behind this is that the protocol is exclusively intended for

² <http://www.aap.org/policy/hyperb.htm>

³ Defined as 37 completed weeks of gestation.

⁴ <http://www.guideline.gov>

the management of jaundice in healthy newborns. An important part of the protocol is the table used to determine the adequate treatment from the TSB value and the age of the infant.

3 Modelling the Jaundice Protocol in Asbru

A number of languages have been proposed to represent medical protocols and their specific features (see [8]). Most protocol-based systems consider protocols as a composition of actions to be performed and conditions to control these actions [9]. Most of them provide some support for text-based or graphical editing of protocols, text annotation of protocols, and simple protocol execution. However, although the trend is changing lately, many of the protocol representation languages in the literature (e.g. GLIF [10]) are not formal enough. For instance, they often incorporate many free-text elements which do not have clear enough semantics. Exceptions to this are PROforma [11] and Asbru [12]. In this work we have chosen Asbru, firstly because it is more precise in the description of various medical aspects, and secondly because Asbru protocols are more declarative, and thus they are more amenable to formal analysis.

3.1 Asbru: A Knowledge Representation Language for Protocols

The main aspects of Asbru are: (i) in Asbru a medical protocol is considered as a plan skeleton with sub-plans in the sense of the AI planning literature, (ii) the possibility to specify the intentions of a plan in addition to the actions of a plan, (iii) the possibility to specify a variety of control-structures within a plan, and (iv) a rich language to specify time annotations. Asbru allows us to represent medical protocols in a precise way. Below we will give a short description of the Asbru language (see [12] for a more detailed description).

Plan A medical protocol is considered as a hierarchical plan. The four main components of a hierarchical plan in Asbru are (1) intentions, (2) conditions, (3) effects and (4) plan-body. Furthermore a plan can have arguments, and has the possibility to return a value. Next we will briefly discuss each of these components.

Intentions Intentions are the high-level goals of a plan. Intentions can be given in terms of achieving, maintaining or avoiding a certain state or action. Such states or actions can be intermediate or final (overall). For example, the label “achieve intermediate-state” means that sometime during the execution of the plan, a certain state must be achieved. “Achieve overall-state” means that at the end of the plan execution, a certain state must be achieved (e.g. at the end of the plan execution, bilirubin levels must be normal). In the same way, “achieve intermediate-action” means that sometime during the plan execution, a certain action must have occurred (e.g. the bilirubin level must have been measured). Notice that in total there are twelve possible forms: [achieve/maintain/avoid] [intermediate/overall]-[state/action].

Conditions There are a variety of conditions that can be associated with an Asbru plan, each of which determines a different aspect of a medical protocol. Asbru has the following conditions:

- filter conditions: These must be true before the plan can be started. For instance, “the blood-type of the mother is not known”.
- setup conditions: Like the filter conditions, these must also be true before the plan can be started, but in this case they can be achieved with additional actions, i.e. by executing other plans.
- suspend conditions: When these are true, the plan will be suspended.
- reactivate conditions: These conditions determine when a suspended plan has to be restarted.
- abort conditions: Such conditions determine when a started, suspended or restarted plan must be aborted.
- complete conditions: These conditions determine when a started or restarted plan can be considered successfully completed.
- activate conditions: These can have the values “manual” or “automatic”. If the activate mode is manual, the user is asked for a confirmation before the plan can be started.

Effects Effects describe the expected effect of a plan on the values of observable medical parameters (e.g. administration of insulin decreases the blood glucose levels). Effects can have associated a likelihood to state how likely they are to occur.

Plan-body The plan-body contains the actions and/or sub-plans that will be executed as part of the plan. A plan-body can have one of the following forms:

- user-performed: an action to be performed by the user, which requires user interaction and thus is not further modelled.
- single step: an action which can be either an activation of a sub-plan, an assignment of a variable, a request for an input value or an if-then-else statement.
- subplans: a set of plan steps to be performed in a given manner. The possibilities are: in sequence (SEQUENTIALLY), in parallel (PARALLEL), in any possible sequential order (ANY-ORDER), and in any possible order, sequential or not (UNORDERED).
- cyclical plan: a repetition of actions over time periods.
- loop construct: a repetition of actions, either in the form of the for loop of conventional programming languages, or iterating on the elements of a list or set.

In the case of subplans, besides the specification of the ordering (SEQUENTIALLY and so forth), it is necessary to specify a waiting strategy. The main aspect here is the so called continuation specification, which describes the plans that must be completed so that the parent plan can be considered successfully completed. For instance, it is possible to define whether all the subplans should be executed (ALL) or not (e.g. ONE or NONE).

Time-annotations Many elements in an Asbru plan (intentions, conditions, effects and plan activations) can have a time annotation. A time annotation specifies (1) in which interval things must start, (2) in which interval they must end, (3) their minimal and maximal duration, and (4) a reference time-point. Any of these elements can be left undefined, allowing the specification of incomplete time annotations. The general scheme for a time annotation is:

([earliest-starting-time, latest-starting-time],
 [earliest-finishing-time, latest-finishing-time],
 [minimal-duration, maximal-duration],
 reference-point)

The use of a time annotation in the context of a plan activation determines the span of time and duration that the plan under execution should have. For example, the action follow a folic acid treatment for 3-4 months, starting in first month of pregnancy, could be expressed as:

Folic-acid-treatment ([week 0, week 4], "start in the first month"
 [week 12, week 20], "end in 3rd, 4th or 5th month"
 [12 weeks, 16 weeks], "do it for 3-4 months"
 conception) "counting from conception"

However, time annotations associated to conditions indicate the period of time during which the conditions are to be evaluated. Once this time has elapsed, there is no possibility for the condition to become true. In case it is necessary to monitor continuously a condition, a special time annotation can be used: NOW.

3.2 Asbru Model of Jaundice Protocol

Figure 2 shows the global structure of the jaundice protocol as a hierarchy of plans. The most important entry point of the protocol is the plan "Diagnostics-and-treatment-hyperbilirubinemia" (the three "Check-for-..." plans are Asbru artifacts to model check-ups at temporally specified intervals). Figure 2 shows that "Diagnostics-and-treatment-hyperbilirubinemia" is divided into a diagnostics and a treatment subplan, to be executed sequentially.

The diagnostics stage is again subdivided into four sequential subplans. One of these plans is "Jaundice-determination", which has four optional subplans among which one of them is required. The protocol specifies that one of the corresponding methods has to be applied to determine if jaundice is clinically significant. This has been modelled as an any-order plan with a waiting strategy ONE, which enables the execution of any of the subplans and states that only one of them is needed. In addition, each subplan has a manual activate condition which requires a confirmation by the user and thus enforces a manual selection.

The treatment phase consists of two subplans (see label (-)): "Regular-treatments" and "Exchange-transfusion". One of them, the "Regular-treatments" plan, contains the main treatment procedure. However, it is possible that this procedure is aborted at some point (when its abort condition becomes true), at which point the "Exchange-transfusion" plan is triggered: it is the emergency action to be taken when the "Regular-treatments" plan aborts. In such an emergency case, the prescriptions of both intensive phototherapy and exchange transfusion must be done, in parallel. In parallel with the treatment plans in group (-), further cyclical actions specify that two important parameters must be measured every 12-24 hours.

The "Regular-treatments" plan has also a quite complicated control structure. This plan consists of two parallel parts: the study of feeding alternatives and the different

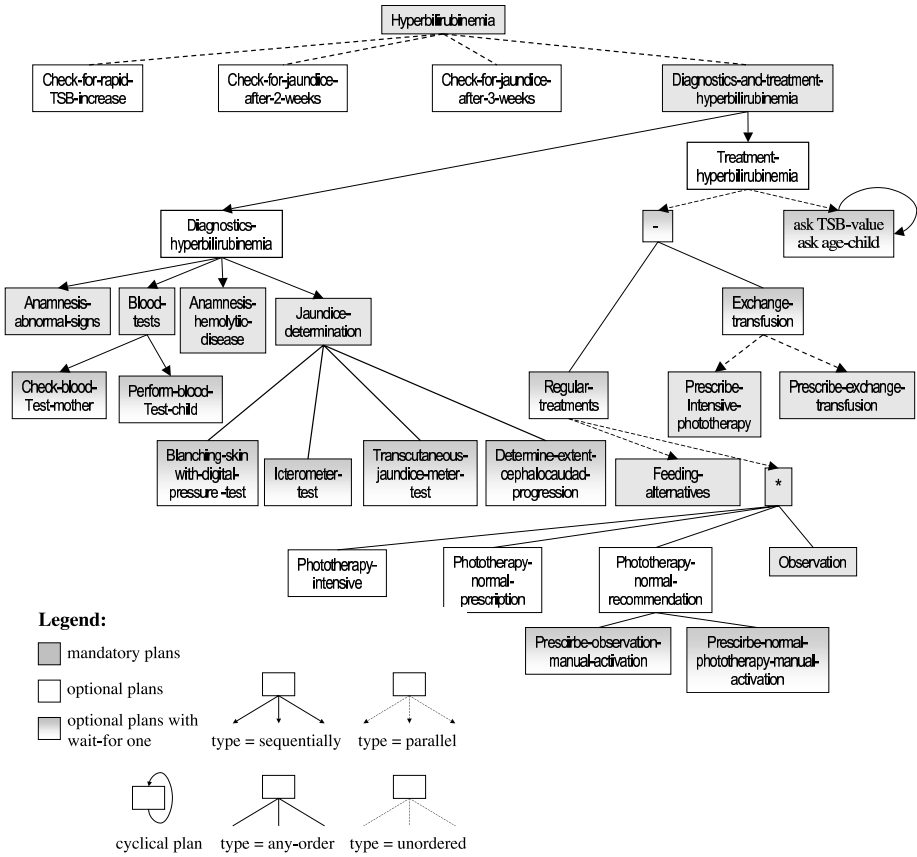


Fig. 2. Overview of the jaundice model in Asbru.

therapies (see label (*)). The plans in group (*) can be tried in any order, one at a time. The intentions of “Regular-treatments” plan are both avoiding toxic bilirubin levels and attaining normal (observation) ones at the end. The plan completes when the feeding alternatives and the therapies complete. The latter in turn depends on the completion of observation (compulsory). It aborts when either bilirubin raises to transfusion levels or intensive phototherapy fails to reduce them sufficiently, pointing to a pathologic reason.

The main surprise from this description is the richness and complexity of the control structures that are found in a medical protocol like the jaundice one: steps are executed in parallel or sequentially, in either a specific or an unspecified order; some steps are compulsory and other steps are optional; some plans are triggered when other plans abort; etc. The Asbru language contains a rich set of modelling primitives to represent these complicated control structures. Notice that these control structures (which apparently appear naturally in a realistic medical protocol) are much more complex than those found in typical programming languages or in planning languages.

The full Asbru specification of the jaundice protocol as well as a high-level overview of its structure can be found in [13]. To give a flavour of Asbru, figures 3 and 4 show, respectively, the “Diagnostics-hyperbilirubinemia” and the “Treatment-hyperbilirubinemia” plans. Notice that the notation used in these figures does not correspond to the XML syntax of the Asbru language, but it is a more readable representation⁵.

PLAN	Diagnostics-hyperbilirubinemia
INTENTIONS	ACHIEVE OVERALL-STATE: is-known(pathologic-reason) AND is-known(jaundice-clinically-significant) NOW
PLAN-BODY	DO type=SEQUENTIALLY, wait-for ALL pathologic-reason = no Anamnesis-abnormal-signs Blood-tests Anamnesis-hemolytic-disease Jaundice-determination

Fig. 3. Plan “Diagnostics-hyperbilirubinemia”.

PLAN	Treatment-hyperbilirubinemia
INTENTIONS	AVOID INTERMEDIATE-STATE: bilirubin = toxic ACHIEVE OVERALL STATE: bilirubin = observation
PLAN-BODY	DO type=PARALLEL, wait-for ONE DO type=ANY-ORDER, wait-for ONE Regular-treatments ON-ABORT Exchange-transfusion Exchange-transfusion CYCLICAL-PLAN DO type=SEQUENTIALLY, wait-for ALL ask TSB-value ask age-child retry-delay min = 12 h, max = 24 h

Fig. 4. Plan “Treatment-hyperbilirubinemia”.

3.3 Benefits of Asbru Modelling: Detection of Protocol Anomalies

During the Asbru formalisation of this protocol, numerous anomalies became apparent. In a general sense, we have used the term anomaly to refer to any issue preventing a satisfactory interpretation of the original protocol. Below we give examples of the different types of anomalies we found. For presentation purposes we have grouped them into three general categories: ambiguity, incompleteness, inconsistency and redundancy.

⁵ The full XML version of the protocol can be found in <http://www.protocure.org/>.

Examples of ambiguity: A problem we encountered during our modelling exercise in jaundice was determining whether the terms “jaundiced”, “clinically jaundiced” and “clinically significant jaundice by medical judgement” have the same meaning or not. These are terms that are used in the flowchart form of the original protocol, but not defined elsewhere. In the Asbru protocol these different terms are translated into the single variable “jaundice-clinically-significant”. See, for instance, the intentions of plan “Diagnostics-hyperbilirubinemia” in figure 3.

Examples of incompleteness: An example of incompleteness anomaly is the following: the original protocol contains a table with “factors to be considered when assessing a jaundiced infant”. One of these factors is “Rapid increase in the TSB level after 24-48 h”. However, what “rapid” exactly means is missing in the protocol. We have solved this problem by looking for the information in other protocols, and have given the rate value 0.5 mg/dl/h. This value is used e.g. in the filter condition of plan “Check-for-rapid-TSB-increase” (see figure 5).

PLAN	Check-for-rapid-TSB-increase
INTENTIONS	ACHIEVE OVERALL-STATE: is-known(possibility-of-G6PD) AND is-known(possibility-of-hemolytic-disease)
CONDITIONS	Filter: (TSB-decrease = no) NOW AND (TSB-change > 0.5) NOW
PLAN-BODY	DO type=SEQUENTIALLY, wait-for ALL possibility-of-hemolytic-disease = yes IF age = day2 THEN possibility-of-G6PD = yes Exit-possibility-of-G6PD ELSE possibility-of-G6PD = no Exit-possibility-of-hemolytic-disease

Fig. 5. Plan “Check-for-rapid-TSB-increase”.

The rate of TSB increase is important for the treatment. The guideline says “Determination of the rate of rise of TSB and the infants age may help determine how often to monitor bilirubin levels and whether to begin phototherapy”. To solve the imprecision of this sentence, we interviewed an expert, who provided us with the information that this monitoring should be done every 12-24 hours. This can be seen in the retry delay specification of the cyclical part within “Treatment-hyperbilirubinemia” plan (see figure 4).

Examples of inconsistency: We found an inconsistency concerning the applicability of the guideline. The guideline is meant for “healthy newborns” according to the title. The protocol specifies that “clinically jaundiced children of ≤ 24 hours old are not considered healthy”. However, elsewhere in the protocol (in point 5 of the Evaluation part), an action is advised for exactly these children (i.e. the children to whom the protocol is not supposed to be applied): “A TSB level needs to be determined in infants noted to be jaundiced in the first 24 hours of life”.

The previous inconsistency occurs only in the text version of the guideline and not in the flowchart form, where a simple exit condition is specified for these children. Our Asbru protocol models this version of the guideline.

Redundancy: Another type of anomaly is redundancy. We did not find any occurrence of this type of anomaly in the jaundice protocol. However, we did find redundancies during the Asbru modelling of a protocol for the management of diabetes mellitus type 2, developed by the Dutch Association of General Practitioners⁶ [14].

To give a better idea of the extent of uncovered anomalies, some numbers follow (see [15] for more details and examples). In the case of jaundice protocol, we found 1 ambiguity, 10 incompleteness anomalies, 6 inconsistencies and no redundancy. Regarding the diabetes protocol, we identified 4 ambiguities, 38 incompletenesses and 2 redundancies, but no inconsistency.

3.4 Experiences and Difficulties

Next we summarise the lessons learned during the modelling of the informal guideline as an Asbru protocol. First of all, not all of Asbru's features described in section 3 were needed to model the protocol. This experience has been confirmed after the modelling other protocols. In particular, the following Asbru constructs were never used: setup, suspend and reactivate conditions, and effects. This has led us to the definition of Asbru-Light, a strict subset of Asbru containing only the features used in our case-studies until now.

Secondly, it was a significant surprise for us that even high quality protocols such as the jaundice protocol of the AAP contain significant numbers of anomalies, including serious problems such as inconsistencies. This already proves that the first step of our formalisation process is worth the significant effort it takes.

Although it is not described in this paper, we have used an interpreter of Asbru-Light to “debug” the jaundice protocol: by running the interpreter on case-data we could check if the protocol behaved as intended. It turns out that using the interpreter is necessary for improving the Asbru model. Of course, such a debug-run cycle is only possible after the protocol has been sufficiently formalised.

A final observation is the significant increase in size when going from the informal, original version of the protocol to the formal version thereof. The original 10 pages of the AAP protocol turned into 40 subplans, taking about 18 pages in the intermediate notation used in the figures above, and more than 2000 lines of XML in the machine readable version of Asbru. We have observed the same effect in our other case-studies.

4 Formalising the Jaundice Protocol in KIV

In the second stage of our formalisation case-study we have used the KIV verification tool [16]. KIV is an interactive theorem prover with strong proof support for higher order logic and elaborate heuristics for automation. Currently, special proof support for temporal logic and parallel programs is being added. In contrast to fully automatic

⁶ <http://nhg.artsennet.nl/standaarden/M01/start.htm>

verification tools, the use of KIV interactive tool allows for the verification of large and complex systems, as it has been shown by its application to a number of real-world systems (distributed systems, control systems, etc.).

4.1 KIV

KIV supports the entire software development process, i.e. the specification, the implementation and the verification of software systems. Next we will briefly describe the relevant aspects of KIV for Asbru specification and verification needs.

For specification, three aspects are important: specifications can be structured, and both functional and operational system aspects can be described. A specification is broken down into smaller and more tractable components using structuring operations such as union and enrichment, that can be used to combine more simple specifications. For functional aspects, algebraic specifications are used to specify abstract data types.

Complex operational behaviour can be specified using parallel programs. Programs in KIV can contain assignments ($v := \tau$), conditionals (**if** φ_{pl} **then** ψ_1 **else** ψ_2), loops (**while** φ_{pl} **do** ψ), local variables (**var** $v = \tau$ **in** ψ), nondeterministic choices (**choose** φ **or** ψ), interleaving ($\varphi \parallel \psi$) and synchronisation points (**await** φ_{pl}).

For a better support of Asbru, additional basic constructs have been implemented: interrupts (**break** ψ **if** φ_{pl}), for modelling different plan conditions, and synchronous parallel execution ($\varphi \parallel_s \psi$), as well as any-order execution ($\varphi \parallel_a \psi$), for a more direct translation of plan bodies. With the help of these constructs, the main features of Asbru-Light can be translated one to one. Others still need to be encoded using additional program variables.

Concerning the verification, we use a variant of Interval Temporal Logic (ITL) [17] to formulate properties about Asbru plans. This logic is first order and allows finite and infinite intervals. In this paper we will restrict ourselves to the temporal operators always (2φ), eventually (3φ), next ($\circ \varphi$), and **last**—which is true only in the last step of an interval.

Single transitions are expressed as first order relations between unprimed and primed variables (v and v'). A primed variable represents the value of the variable in the next state. For example, the formula $v = 0 \wedge (2 v' = v + 1) \rightarrow 3 v = n$ states that, if variable v is initially 0, and the value v' in the next state is always incremented by one, then eventually the variable will be equal to an arbitrary natural number n .

In KIV, the proof technique for verifying parallel programs is symbolic execution with induction. Details can be found in [18]. Since programs are treated as formulas—for both, the semantics is a set of traces—they can be arbitrarily mixed. This gives rise to a modular proof technique, which is very important for the verification of Asbru plans as they tend to be large.

4.2 KIV Formalisation of Jaundice Protocol

In order to formally examine Asbru plans in a first attempt, we have translated them into parallel programs. The translation of the Asbru model into KIV has been done in a structure preserving way, by mapping each Asbru plan into a KIV specification. This

has been possible thanks to the modularisation facilities that KIV provides. Thus, the structure of the jaundice protocol in KIV roughly mirrors the Asbru model shown in figure 2. This is one of the key ideas of our formalisation strategy, because it gives the possibility to obtain some feedback from the specification and verification phases in terms of the Asbru model, and to exploit this structure during proof attempts.

Following this idea, Asbru plans have to be translated into different types of KIV programs. For the moment this translation has been performed manually. Table 1 gives some of the translations of Asbru constructs into KIV programs that we have used in this process.

Table 1. Translation of some Asbru constructs into KIV.

Asbru	KIV
filter condition φ NOW body	await φ ; body
filter condition φ body	if φ then body
complete condition φ body	break body if φ
abort condition φ body	break body if φ
$\langle\langle\text{name}\rangle\rangle$ (<i>plan activation</i>)	$\langle\langle\text{name}\rangle\rangle\#(\dots)$ (<i>procedure call</i>)
do type=sequentially P1,... Pn	P1;... Pn
do type=any-order P1,... Pn	P1 \parallel_a ... Pn
do type=unordered P1,... Pn	P1 \parallel_s ... Pn
wait-for Pi body	break body if <i>some expression on Pi-state</i>

The example in figure 6, corresponding to the plan “Diagnostics-hyperbilirubinemia” of figure 3, serves to illustrate the kind of translations that have been obtained. In this example we can see that the KIV translation closely follows the structure of the original Asbru plan, except for an additional interrupt (break) construct. This construct has been introduced to model the waiting strategy of the plan, which is “wait-for ALL”. This implies that all the subplans must complete successfully so that the parent plan can do so. Conversely, as soon as any of the subplans abort, the parent will abort too. The latter has been modelled with the help of specific plan state variables which are explicitly set within the subplans. Other translations, however, did not result in a version so close to the Asbru plan. This is due to the encodings necessary to represent the Asbru elements not directly supported by KIV.

Currently we are working on the verification of several protocol properties. Properties are expressed in the above described variant of ITL. For instance:

$$Diagnostics-hyperbilirubinemia\#(\dots) \wedge (2\ time'' = time' + 1) \rightarrow 3\ \mathbf{last}$$

is a property expressing the termination of the previous program/Asbru plan. It states that, if the program *Diagnostics-hyperbilirubinemia* is executed, it will stop some-time in the future. The always formula in the antecedent is used to model the environment, in which time changes from one state to the next.

Termination of (sub)plans is a basic property necessary to prove the termination of the protocol. Although it might seem a not very interesting property in itself, our

```

Diagnostics-hyperbilirubinemia#(var patient-data, time,
  jaundice-clinically-significant, pathologic-reason)
begin
  var anamnesis-abnormal-signs-state = inactive,
      blood-tests-state = inactive,
      anamnesis-hemolytic-disease-state = inactive in begin
  break begin
    pathologic-reason := false;
    anamnesis-abnormal-signs#(; time, pathologic-reason,
      anamnesis-abnormal-signs-state);
    blood-tests#(; patient-data, time, pathologic-reason,
      blood-tests-state);
    anamnesis-hemolytic-disease#(; time, pathologic-reason,
      anamnesis-hemolytic-disease-state);
    jaundice-determination#(; time, jaundice-clinically-significant)
  end
  if anamnesis-abnormal-signs-state = aborted
    ∨ blood-tests-state = aborted
    ∨ anamnesis-hemolytic-disease-state = aborted
  end
end

```

Fig. 6. KIV translation of “Diagnostics-hyperbilirubinemia” plan.

experiences until now show that it can serve to identify assumptions implicitly made in the protocol. These assumptions could be used e.g. to improve the description of the applicability conditions of the original protocol.

Another promising property is ensuring that the intentions of a plan follow from the intentions of its subplans. Although Asbru intentions are not included in the KIV formulation, they can be used to verify if the composition of subplans complies with what is intended in the plan.

As part of the IST Protocure⁷ project, we are investigating other properties more significant from the medical point of view. Amongst them, we can cite the use of indicators issued by medical organisations, which define a variety of features that specific protocols should comply with.

4.3 Experiences and Difficulties

In the following paragraphs we briefly describe the experiences in the second stage of our formalisation case-study, and in our first verification attempts.

First, the KIV formalisation step has taken considerably less effort than the Asbru modelling one. This is mainly due to the structure preserving strategy we have followed. Thanks to it, the formalisation roughly consists in the translation of Asbru plans into KIV procedures.

Second, concerning this translation, a limitation of the current approach is that it is not automatic. Besides, in some cases it requires many creative tricks to adequately

⁷ <http://www.protocure.org/>

encode the Asbru constructs not directly supported by KIV. As result, sometimes the KIV translation suffers from a weak resemblance to the initial Asbru protocol. These problems will be solved if verification turns out to be profitable, by means of a direct KIV support of Asbru syntax allowing for a direct translation of arbitrary Asbru models.

We cannot strictly say that the formalisation in KIV has contributed to the improvement of the original protocol, as in the case of its Asbru modelling. As for the verification, after the completion of the first proofs we can say that it is feasible and that it serves to detect implicit knowledge, such as underlying assumptions. We are confident that it is possible to use the jaundice formalisation and KIV for the verification of more significant properties like the ones mentioned before, which could actually be used to improve the original protocol.

5 Conclusions

It is of course well known that many forms of knowledge can be represented in languages that are formalised to a certain extent. Indeed this is the entire premise of fields such as knowledge engineering and knowledge representation. However, we would argue that it is a non-trivial result that it has turned out to be feasible to formalise a significant piece of realistic medical knowledge to such an amount of detail that it can be used as the basis for mechanised theorem proving (a much greater level of formality than is used even in common mathematical publications). This shows that the traditional gap between practical knowledge engineering and academic formal knowledge representation can indeed be bridged even for realistic applications.

Naturally, such an achievement comes at a price: a significant amount of effort is required for such a formalisation effort. Although we are not in a position to make strong quantitative statements, the case-study reported has taken close to a person-year to complete.

However, we would argue that this price is worth paying. A number of anomalies were uncovered in the original medical guideline, even though this guideline is representative of the best quality that the medical profession can offer. All of these anomalies were uncovered in the first stage of our formalisation (from original guideline to Asbru). The most important contribution of the second stage of our formalisation (from Asbru to KIV) until now has been to disambiguate any remaining unclarities in the Asbru model that resulted from the first stage: a number of semantic problems with Asbru were uncovered, and finally resolved by providing a fully formal semantics of Asbru in KIV. To date, we have only very limited experience in using the KIV formalisation in formal proofs of properties of the protocol. We expect that this usage of KIV will uncover further anomalies in the protocol.

A final observation is that of the two steps in our formalisation process (from original guideline to Asbru, and from Asbru to KIV), the first step was by far the most labour intensive. This step involved most of the conceptual analysis that was required for the formalisation. Consequently, we would argue that this stage of the formalisation process would benefit from being split up in a number of smaller steps, each yielding its own model, in ever increasing degrees of formality.

Acknowledgements

This work has been partially supported by the European Commission's IST program, under contract number IST-2001-33049–Protocure. We also want to thank Hugo Roomans and Geert Berger, for their contribution to the Asbru modelling of the jaundice protocol, Tibor Bosse, for his work on the interpreter of Asbru-Light and his efforts in debugging the protocol, and all other Protocure members: Silvia Miksch, Andreas Seyfang, Wolfgang Reif, Cristoph Duelli, Kitty Rosenbrand, Joyce van Croonenborg, and Peter Lucas.

References

- [1] Weingarten, S.: Using Practice Guideline Compendiums To Provide Better Preventive Care. *Annals of Internal Medicine* **130** (1999) 454–458
- [2] Field, M., Lohr, K., eds.: *Clinical Practice Guidelines: Directions for a New Program*. National Academy Press, Washington D.C., USA (1992)
- [3] Woolf, S., Grol, R., Hutchinson, A., Eccles, M., Grimshaw, J.: Potential benefits, limitations, and harms of clinical guidelines. *British Medical Journal* **318** (1999) 527–530
- [4] Clayton, P., Hripsak, G.: Decision support in healthcare. *Int. J. of Biomedical Computing* **39** (1995) 59–66
- [5] Graham, I., Calder, L., Hébert, P., Carter, A., Tetroe, J.: A comparison of clinical practice guideline appraisal instruments. *International Journal of Technology Assessment in Health Care* **16** (2000) 1024–1038
- [6] Musen, M., Rohn, J., Fagan, L., Shortliffe, E.: Knowledge engineering for a clinical trial advice system: Uncovering errors in protocol specification. *Bulletin du Cancer* **74** (1987) 291–296
- [7] AAP: American Academy of Pediatrics, Provisional Committee for Quality Improvement and Subcommittee on Hyperbilirubinemia. Practice parameter: management of hyperbilirubinemia in the healthy term newborn. *Pediatrics* **94** (1994) 558–565
- [8] Elkin, P., Peleg, M., Lacson, R., Bernstam, E., Tu, S., Boxwala, A., Greenes, R., Shortliffe, E.: Toward Standardization of Electronic Guidelines. *MD Computing* **17** (2000) 39–44
- [9] Miksch, S.: Plan Management in the Medical Domain. *AI Communications* **12** (1999) 209–235
- [10] Ohno-Machado, L., Gennari, J., Murphy, S., Jain, N., Tu, S., Oliver, D., Pattison-Gordon, E., Greenes, R., Shortliffe, E., Octo Barnett, G.: Guideline Interchange Format: a model for representing guidelines. *J. of the American Medical Informatics Association* **5** (1998) 357–372
- [11] Fox, J., Johns, N., Lyons, C., Rahmzadeh, A., Thomson, R., Wilson, P.: PROforma: a general technology for clinical decision support systems. *Computer Methods and Programs in Biomedicine* **54** (1997) 59–67
- [12] Shahar, Y., Miksch, S., Johnson, P.: The Asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines. *Artificial Intelligence in Medicine* **14** (1998) 29–51
- [13] Roomans, H., Berger, G., Marcos, M., ten Teije, A., Seyfang, A., van Harmelen, F.: *Asbru Protocol for the Management of Hyperbilirubinemia in the Healthy Term Newborn*. Technical Report IR-495, Vrije Universiteit Amsterdam (2002) To be published.
- [14] Rutten, G., Verhoeven, S., Heine, R., de Grauw, W., Cromme, P., Reenders, K., van Ballegoie, E., Wiersma, T.: NHG-Standaard Diabetes Mellitus Type 2 (eerste herziening). *Huisarts en Wetenschap* **42** (1999) 67–84 First revision.

- [15] Marcos, M., Roomans, H., ten Teije, A., van Harmelen, F.: Improving medical protocols through formalisation: a case study. In: Session on Formal Methods in Healthcare, 6th International Conference on Integrated Design and Process Technology (IDPT-02). (2002)
- [16] Balsler, M., Reif, W., Schellhorn, G., Stenzel, K., Thums, A.: Formal system development with KIV. In Maibaum, T., ed.: *Fundamental Approaches to Software Engineering*. Number 1783 in LNCS, Springer (2000)
- [17] Moszkowski, B.: A temporal logic for multilevel reasoning about hardware. *IEEE Computer* **18** (1985) 10–19
- [18] Balsler, M., Duelli, C., Reif, W., Schellhorn, G.: Verifying concurrent systems with symbolic execution. *Journal of Logic and Computation (Special Issue)* (2002)