in: Fourth European Conference on Planning ECP '97 (EWSP '97), Springer, Lecture Notes in Artificial Intelligence (LNAI), Toulouse, France, September 24-26,1997.

Time-Oriented Skeletal Plans: Support to Design and Execution

Silvia Miksch[1], Yuval Shahar[2], Werner Horn[3], Christian Popow[4], Franz Paky[5], Peter Johnson[6]

[1] Vienna University of Technology, Institute of Software Technology, Resselgasse 3/188, A-1040 Vienna, Austria, email: silvia@ifs.tuwien.ac.at

[2] Section on Medical Informatics, Stanford University, Stanford, CA 94 305 - 5479, USA email: shahar@smi.stanford.edu

[3] Austrian Research Institute for Artificial Intelligence (ÖFAI) & Department of Medical Cybernetics and Artificial Intelligence, University of Vienna, Freyung 6, A-1010 Vienna, Austria, email: werner@ai.univie.ac.at

[4] NICU, Department of Pediatrics, University of Vienna, Währinger Gürtel 18-20, A-1090 Vienna, Austria, email: popow@vm.akh-wien.ac.at

[5] Department of Pediatrics, Hospital of Mödling, Weyprechtgasse 12, A-2340 Mödling, Austria, email: franz.paky@magnet.at

[6] The Sowerby Unit for Primary Care Informatics, University of Newcastle, 21 Claremont Place, Newcastle Upon Tyne, NE2 4AA, UK, email: pete@mimir.demon.co.uk

Abstract. Skeletal plans are a powerful way to reuse existing domain-specific procedural knowledge. In the **Asgaard** project, a set of tasks that support the design and the execution of skeletal plans by a human executing agent other than the original plan designer are designed. The underlying requirement to develop task-specific problem-solving methods is a *modeling languag*e. Therefore, within the Asgaard project, a time-oriented, intention-based language, called **Asbru**, was developed. During the design phase of plans, Asbru allows to express durative actions and plans caused by durative states of an observed agent. The intentions underlying these plans are represented explicitly as temporal patterns to be maintained, achieved or avoided. We will present the underlying idea of the Asgaard project and explain the time-oriented Asbru language. Finally, we show the benefits and limitations of the time-oriented, skeletal plan representation to be applicable in real-world, high-frequency domains.

1 Motivation and Introduction

We are motivated by the need for knowledge-based support in the medical domain. Health care providers are faced with two problems: (1) the information overload resulting from modern equipment, and (2) improving the quality of health care through increased awareness of proper disease management techniques. *Clinical protocols* and *guidelines* should solve the difficulties. Clinical guideline refers to a general principle by which a course of actions is determined and clinical protocol refers to a general class of therapeutic interventions. In the following, we will use clinical guideline and protocol interchangeable.

Appropriate clinical protocols are only available for a very limited class of clinical problems. They are not adjusted to the patient data-management system, they are partly vague and incomplete concerning their intentions and their temporal and context-dependent representation, and most often they are outdated after being developed. Extracting and formulating the knowledge structure for clinical protocols is a non trivial task. The context implicit in the protocols must be made explicit.

1.1 Automated Support to Protocol-Based Care

During the last few years, there have been several efforts to create automated reactive planners to support the process of protocol-based care over periods of time. In the *prescriptive* approach, active interpretation of the guidelines is given; examples include ONCOCIN (Tu et al. 1989) in the oncology domain and the DILEMMA project (Herbert et al. 1995), the EON architecture (Musen et al. 1996), the PROMPT project (Fox et al. 1997) and the PRESTIGE project (Gordon et al. 1997), as general architectures. In the *critiquing* approach, the program critiques the physician's plan rather than recommending a complete one of its own. This approach concentrates on the user's needs and assumes that the user has considerable domain-specific knowledge (Miller 1986).

Several approaches to the support of guideline-based care encode guidelines as elementary state-transition tables or as situation-action rules dependent on the electronic medical record (Sherman et al. 1995), but do not include an intuitive representation of the guideline's clinical logic, and have no semantics for the different types of clinical knowledge represented. Other approaches permit hypertext browsing of guidelines via the World Wide Web (Barnes and Barnett 1995), but do not use the patient's electronic medical record.

The most favored attempts capturing and supporting clinical procedures, are *flow diagrams* and *flowcharting tools*. The medical experts are mostly used to working with these techniques. However, it is quite difficult to cope with all possible orders of plan execution and all the exception conditions that might arise. Clinical protocols are a way of pre-compiling decisions that must be made, in which experts knowledge is distilled into a form of procedural knowledge. The trouble is that this by necessity can only cover a small subset of the possible situations and possible paths through. Additionally, medical experts have a lot of difficulties to define their metrics for measuring the success or failure of an individual action. The best experts often have their own personal metrics by which they judge the success or failure of an action they have taken. But these metrics are usually very arbitrary, based on empirical factors, and difficult to extract from the expert. They may differ from one expert to another quite widely. The expert cannot usually be pushed into providing the evidence for these metrics.

1.2 Skeletal Plans

A common strategy for the representation and the reuse of domain-specific procedural knowledge is the representation of that knowledge as a library of skeletal plans. Skeletal plans are plan schemata at various levels of detail that capture the essence of the procedure, but leave room for execution-time flexibility in the achievement of particular (Friedland and Iwasaki 1985). Thus, they are usually reusable in different contexts. The idea was proposed to reduce complexity of planning, called skeletal-plan refinement. Instead of planning in an unconstrained search space, the skeletal-plan refinement method relies on available abstraction (or skeletal) plans which were refined in the context of a particular problem. Later similar ideas were exploited to create automated reactive planners, such as ONCOCIN (Tu et al. 1989), SPIN (Uckun 1994), and a KADS model for hierarchical skeletal plan refinement (Aitken and Shadbolt 1994).

1.3 Modeling Languages

On the one hand, workers in medicine and medical informatics have recognized the importance of protocol-based care to ensure a high quality of care since the 1970s. A group of investigators, working through the American Society for Testing and Materials (ASTM), has defined a standard procedural language, known as the Arden syntax (Hripcsak et al. 1994). The Arden syntax encodes situation-action rules. Developers of the Arden syntax have promoted this Pascal-like language because of the pressing needs to facilitate exchange of guidelines among health-care institutions using existing software technology. This standard has significant limitations: The language currently supports only atomic data types, lacks a defined semantic for making temporal comparisons or for performing data abstraction, and provides no principled way to represent clinical guidelines that are more complex than individual situation-action rules (Musen et al. 1995). Therefore the Arden syntax is not applicable for our purposes.

On the other hand, computer-oriented knowledge interchange languages (e.g., KIF (Genesereth and Fikes 1992)), ontologies or models for knowledge sharing (e.g., Gruber (1993); Guarina and Giaretta (1995)), and general purpose languages to support planning (e.g., PROPEL language (Levinson 1995), O-Plan2 (Tate et al. 1994) were introduced. These traditional (plan-execution) representations have significant limitations and are not applicable in dynamic changing environments, like medical domains: (1) they assume instantaneous actions and effects; (2) actions often are continuous (durative) and might have delayed effects and temporally-extended goals (Bacchus and Kabanza 1996); (3) there is uncertainty and variability in the utility of available actions; (4) unobservable underlying processes determine the observable state of the world; (5) a goal may not be achievable; (6) parallel and continuous execution of plans is necessary. The requirements of plan specifications in clinical domains (Tu et al. 1989; Uckun 1994) are often a superset of the requirements in typical toy-problem domains used in planning research.

A sharable skeletal-plan-execution language needs to be expressive with respect to temporal annotations and needs to have a rich set of parallel, sequential, and iterative operators. Thus, it should enable designers to express complex procedures in a manner similar to a real programming language (although typically on a higher level of abstraction). The language, however, also requires well-defined semantics for both the prescribed actions and the task-specific annotations, such as the plan's intentions and effects, and the preferences (e.g., implicit utility functions) underlying them. Thus, the executing agent's (e.g., the physician's) actions can be better supported, leading to a more flexible dialog and, in the case of the clinical domains, to a better acceptance of automated systems for guideline-based care support. Clear semantics for the task-specific knowledge roles also facilitate acquisition and maintenance of these roles.

With these requirements in mind, we have developed a time-oriented, intention-based, and sharable language, called **Asbru**. The Asbru language is part of the **Asgaard** project (Shahar et al. 1996a), in which we are developing task-specific problem-solving methods that perform design, execution, and critiquing tasks in medical domains. (In Norse mythology, Asgaard was the home and citadel of the gods. It was located in the heavens and was accessible only over the rainbow bridge, called Asbru.)

Section 2 gives an overview about the Asgaard project. Section 3 and 4 explains the various components of the time-oriented, intention-based language Asbru. Section 5 characterizes the medical problem of artificial ventilated new-born infants and illustrates how it is represented in Asbru. In the last section we evaluate Asbru's applicability identifying it's strengths and limitations.

2 The Asgaard Project

The **Asgaard** project outlined some useful task-specific problem-solving methods to support both designer and executor of skeletal plans. The project is oriented to support therapeutic issues. The problem-solving methods are divided in tasks, which are performed during *design time* and *execution time* of a skeletal plan (Table 1). Each task can be formulated as answering a specific set of questions. A more detailed description can be found in (Shahar et al. 1996a).

Table 1. Overview of the support tasks during design and execution time

Time	Task	Questions to be answered
Design time	Verification	Are the intended plans compatible with the prescribed actions?
	Validation	Are the intended states compatible with the prescribed actions and intended plans?
Execution time	Applicability of plans	What skeletal plans are applicable this time to this world?
	Execution of plans	What should be done now according to the execution-plan's prescribed actions?
	Recognition of intentions	Why is the executing agent executing a particular set of actions, especially if those actions deviate from the skeletal plan's prescribed actions?
	Critique of the executing agent's actions	Is the executing agent deviating from the prescribed actions or intended plan? Are the deviating actions compatible with the author's plan and state intentions?
	Evaluation of the plan	Is the plan working?
	Modification of an executing plan	What alternative actions or plans are relevant at this time for achieving a given state intention?

3 The Asbru Language: Basic Concepts

Asbru can be used to design specific plans as well as support the performance of different reasoning and executing tasks. During the design phase of plans, Asbru provides a powerful mechanism to express durative actions and plans caused by durative states of an observed agent (e.g., many actions and plans need to be executed in parallel or every particular time point). These plans are combined with intentions of the executing agent of plan. They are uniformly represented and organized in the *guideline-specification library*. During the execution phase an applicable plan is instantiated with distinctive arguments and state-transition criteria are added to execute and reason about different tasks.

3.1 Meaning of "Intention-based"

The meaning of *intentions* in general and for planning tasks in particular has been examined in philosophy (Bratman 1987) and in artificial intelligence (Pollack 1992). We view intentions as temporally extended goals at various abstraction levels (Bacchus and Kabanza 1996). Intentions are *temporal patterns* of actions or states, to be maintained, achieved, or avoided.

3.2 Meaning of "Temporal Pattern" and "Time annotations"

Intentions, world states, and prescribed actions are *temporal patterns*. A temporal pattern is (1) a *parameter proposition*: a parameter (or its abstraction), its value, a context, and a time annotation; (2) a *combination* of multiple parameter propositions (Shahar and Musen 1996b); or (3) a *plan-state* associated to an instantiated plan (plan pointer) and a time annotation.

The *time annotation* we use allows a representation of uncertainty in starting time, ending time, and duration (Dechter et al. 1991; Rit 1986). The time annotation supports multiple time lines (e.g., different zero-time points and time units) by providing *reference annotations*. We define temporal shifts from the reference annotation to represent the uncertainty in starting time, ending time, and duration, namely earliest starting shift (ESS), latest starting shift (LSS), earliest finishing shift (EFS), latest finishing shift (LFS), minimal duration (MinDu), and maximal duration (MaxDu). The temporal shifts are associated with time units (e.g., minutes, days) or domain-dependent units. Thus, our temporal annotation is written as ([ESS, LSS], [EFS, LFS], [MinDu, MaxDu], REFERENCE). Figure 2 illustrates our time annotation. ESS, LSS, EFS, LFS, MinDu, and MaxDu can be "unknown" or "undefined" to allow incomplete time annotation.

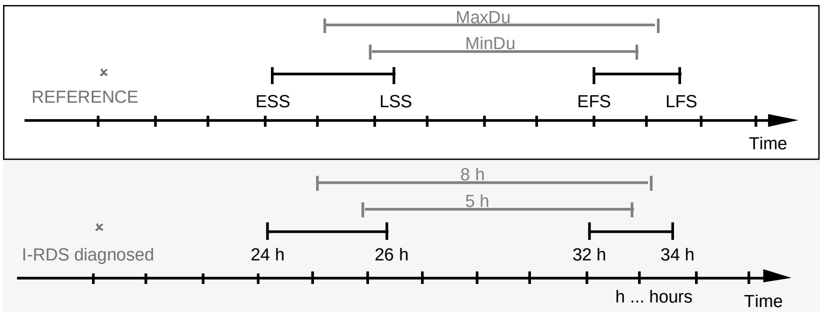

Fig. 2. A schematic illustration of the Asbru time annotations. The upper part of the figure presents the generic annotation. The lower part shows a particular example representing the time annotation [[24 HOURS, 26 HOURS], [32 HOURS, 34 HOURS], [5 HOURS, 8 HOURS], I-RDS-diagnosed]), which means "starts 24 to 26 hours after I-RDS was diagnosed, ends 32 to 34 hours after the I-RDS was diagnosed , and lasts 5 to 8 hours".

For example, the parameter proposition "the level of blood gas is normal or above the normal range in the context of controlled ventilation-therapy for at least three hours, using the activation of the plan as reference point", is written in Asbru as:

```
(STATE(BG) (OR NORMAL ABOVE-NORMAL) controlled-ventilation
       [[  _, _], [_, _], [180 MIN,_], *self*])
```

To allow temporal repetitions, we define sets of cyclic time points (e.g., MIDNIGHTS, which represents the set of midnights, where each midnight occurs exactly at 0:00 A.M., every 24 hours) and cyclic time annotations (e.g., MORNINGS, which represents a set of mornings, where each morning starts at the earliest at 8:00 A.M., ends at the latest at 11:00 A.M., and lasts at least 30 minutes). In addition, we allow certain short-cuts such as for the current time, whatever that time is (using the symbol *NOW*), or the duration of the plan (using the symbol *). Thus, the Asbru notation enables the expression of interval-based intentions, states, and prescribed actions with uncertainty regarding starting, finishing, duration, and the use of absolute, relative, and even cyclical (with a predetermined granularity) reference annotations. All domain-dependent time annotations, units, and time abstractions have to be defined in advance to be applicable in all plans in the guideline-specification library. The definitions ensure that site-specific practice can be clarified and specified (e.g., DAYS start at 0:00 am or DAYS start at 7:00 am). In addition, a sampling-frequency argument specifies the frequency of sampling the external-world's data, such as when verifying the applicability of a particular plan. Thus, we define a sampling frequency for examining the plan's state-transition criteria (see Sect. 3.4).

3.3 Decomposition and "Semantic" Stop-Condition

A *plan* in the guideline-specification (plan) library is composed hierarchically, using the Asbru syntax, of a set of plans with arguments and time annotations. A decomposition of a plan into its subplans is always attempted by the execution interpreter, unless the plan is not found in the guideline-specification library, thus representing a nondecomposable plan (informally, an *action* in the classical planning literature). This can be viewed as a "semantic" stop-condition. Such a plan is referred to the agent for execution, which may result in an interaction with a user or an external calling of a program. Plans have return values.

3.4 Plan States and State-Transition Criteria

During the execution phase, an applicable plan is instantiated. A set of mutually exclusive *plan states* describes the actual status of the plan during execution. Particular *state-transition criteria* specify transition between neighboring plan states. Figure 3 illustrates the different plan states and their corresponding transition criteria mentioned on the arrows. The meaning of the state-transition criteria is explained in Sect. 4.3. We distinguish between plan states during the plan-selection phase (left-hand side of Fig. 3) and between plan states during the execution phase (right-hand side of Fig. 3). For example, if a plan has been `activated`, it can only be `completed`, `suspended`, or `aborted` depending on the corresponding criteria. The gray triangle on the right-hand side of Fig. 3 includes the three basic states; these should always be defined. The `suspended` state is optional and is available for more complex plan types. A `suspended` plan can be either reactivated or restarted. The reactivation depends on the `reactivate` condition. The restart condition is defined implicitly: first, abort an activated plan and then restart it from the `considered` state.

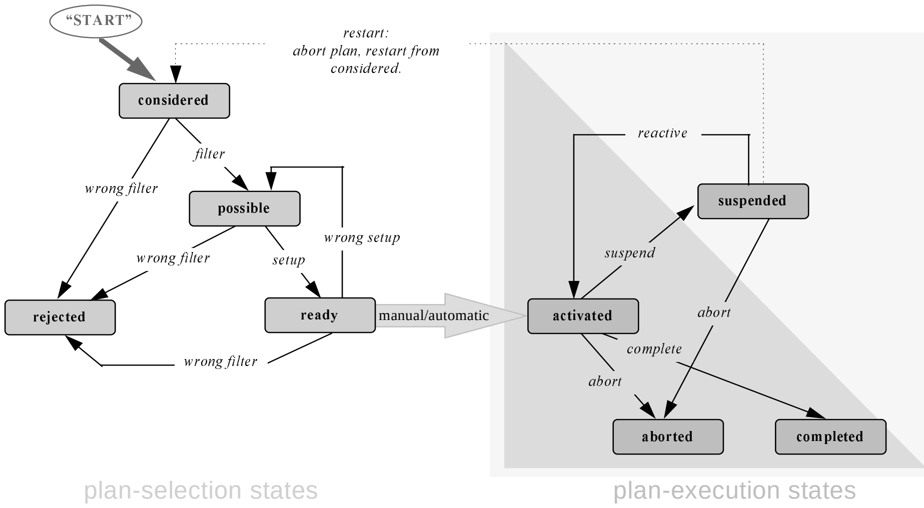

Fig. 3. The plan-instance states and their associated state-transition criteria used in Asbru.

4 Components of Asbru

A plan consists of a name, a set of arguments, including a time annotation (representing the temporal scope of the plan), and five components: **preferences**, **intentions**, **conditions**, **effects**, and a **plan body** which describes the actions to be executed. The general arguments, the time annotation, and all components are optional. A subplan has the same structure (Fig. 4a). An example is given in Fig. 4b, the example is described in Sect. 5.

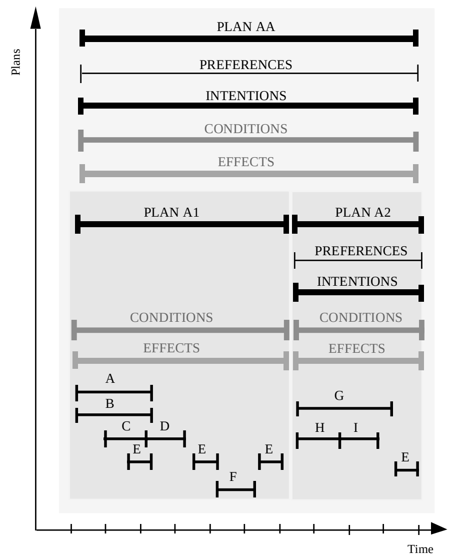

Fig. 4a. Graphical representation of a clinical-guideline specification in Asbru.

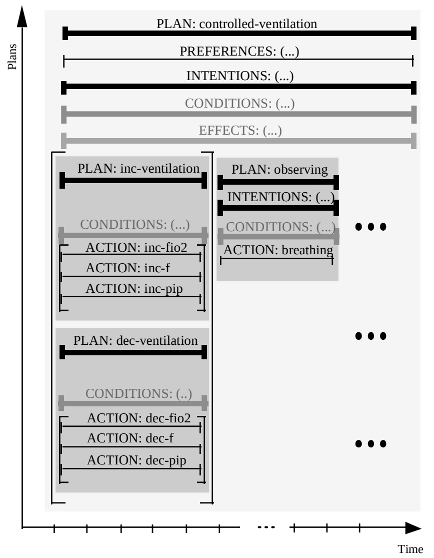

Fig. 4b: Subplans of the treatment protocol for immature respiratory distress syndrome.

4.1 Preferences

Preferences bias or constrain the selection of a plan to achieve a given goal and express a kind of behavior of the plan. We distinguish between:

(1) **Strategy**: a general strategy for dealing with the problem (e.g., aggressive);
(2) **Utility**: a set of utility measures (e.g., minimize the cost or inconvenience);
(3) **Select-method**: a matching heuristic for the applicability of the whole plan (e.g., exact-fit);
(4) **Resources**: a specification of prohibited or obligatory resources (e.g., in certain cases of treatment of a pulmonary infection, surgery is prohibited and antibiotics must be used);
(5) **Start-conditions:** an indication whether transition from a `ready` generic plan to the `started` state of an actual plan instance is automatic (after applying the *filter* and *setup* preconditions—see below) or requires approval of the user.

4.2 Intentions

Intentions are high-level goals at various levels of the plan, an annotation specified by the designer, which supports tasks such as critiquing and modification. Intentions are temporal patterns of executing-agent actions and external-world states that should be maintained, achieved, or avoided. We define four categories of intentions:

(1) **Intermediate-state:** the state(s) that should be maintained, achieved, or avoided during the applicability of the plan (e.g., the blood-gas levels are slightly below to slightly above the target range);
(2) **Intermediate-action:** the action(s) that should take place during the execution of the plan (e.g., minimize level of mechanical ventilation);
(3) **Overall-state-pattern:** the overall pattern of states that should hold after finishing the plan (e.g., patient had less than one high blood-gas value per 30 minutes);
(4) **Overall-action-pattern:** the overall pattern of actions that should hold after finishing the plan (e.g., avoid hand-bagging).

4.3 Conditions

Conditions are temporal patterns, sampled at a specified frequency, that need to hold at particular plan steps to induce a particular state transition of the plan instance. We do not directly determine conditions that should hold during execution. We specify different conditions that enable transition from one plan state into another (see Fig. 3). A plan is completed when the completed conditions become true, otherwise the plan's execution suspends or aborts. Aborting a plan's execution is often due to a failure of the plan or part of it. All conditions are optional. We distinguish between:

(1) **Filter-preconditions** need to hold initially if the plan is applicable, but can not be achieved (e.g., female). They are necessary for a state to become `possible`;

(2) **Setup-preconditions** need to be achieved to enable a plan to start (e.g., inspiratory oxygen concentration F_iO_2 is less than 80%) and allow a transition from a `possible` plan to a `ready` plan;
(3) **Suspend-Conditions** determine when an `activated` plan has to be suspended – certain conditions (*protection intervals)* need to hold (e.g., blood gas has been extremely above the target range for at least five minutes);
(4) **Abort-Conditions** determine when an `activated`, `suspended`, or `reactivated` plan has to be aborted (e.g., the increase of the blood-gas level is too-fast for at least 30 seconds);
(5) **Complete-conditions** determine when an `activated` or `reactivated` plan has to be completed successfully (e.g., returning to spontaneous breathing);
(6) **Reactivate-Conditions** determine when a `suspended` plan has to be `reactivated` (e.g., blood gas level is back to normal or slightly increased).

4.4 Effects

Effects describe the functional relationship between the plan arguments and measurable parameters (e.g., the *dose* of insulin is inversely related to the level of blood glucose) or the overall effect of a plan on parameters (e.g., administration of insulin decreases the blood glucose). Effects have a likelihood annotation—a probability of occurrence.

4.5 Plan-Body

The plan body is a set of plans to be executed in parallel, in sequence, in any order, or in some frequency. We distinguish among several types of plans: *sequential*, *concurrent*, and *cyclical*. Only one type of plan is allowed in a single plan body. A sequential plan specifies a set of plans that are executed in sequence; for continuation, all plans included have to be completed successfully. Concurrent plans can be executed in parallel or in any order. We distinguish two dimensions for classification of sequential or (potentially) concurrent plans: the number of plans that should be completed to enable continuation and the order of plan execution. The continuation condition specifies the names of the plans that must be completed to proceed with the next steps in the plan. A cyclical plan (an EVERY clause) includes a plan that can be repeated, and optional temporal and continuation arguments that can specify its behavior.

5 Treatment Protocols for Artificial Ventilated Newborn Infants

Artificial ventilation has greatly contributed towards the improvement of the mortality and morbidity of premature newborn infants. However, standardized clinical treatment protocols for immature respiratory distress syndrome (I-RDS) are partly vague and incomplete concerning their intentions and their temporal and context-dependent representation. Therefore, we acquired the implicit or not mentioned intentions and conditions from domain experts.

Figure 5 illustrate the top-level treatment protocol for I-RDS. After I-RDS is diagnosed, a plan dealing with limited monitoring possibilities is activated, called initial-phase. Depending on the severity of the disease, three different kinds of plans can follow, controlled-ventilation, permissive-hypercapnia, or crisis-management. Only one plan at a time can be activated, however the order of execution and the activation frequency of the three different plans depend on the severity of the disease. The brackets in Fig. 5 illustrate this. Additionally, it is important to continue with the plan weaning only after a successful completion of the plan controlled-ventilation. After a successful execution of the plan weaning, the extubation should be initiated. The extubation can be either a single plan extubation or a sequential execution of the subplans cpap and extubation.

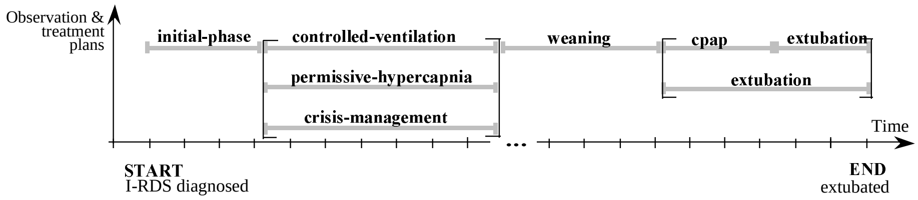

Fig. 5. Treatment protocol for immature respiratory distress syndrome (I-RDS).

The following specification shows the treatment protocol I-RDS in Asbru syntax:

```
(PLAN I-RDS-therapy
   <... parts deleted ...>
(DO-ALL-SEQUENTIALLY
   (initial-phase)
   (one-of-controlled-ventilation)
   (weaning)
   (one-of-cpap-extubation)))
```

```
(PLAN one-of-controlled-ventilation
   <... parts deleted ...>
(DO-SOME-ANY-ORDER
   (controlled-ventilation)
   (permissive-hypercapnia)
   (crisis-management)
   CONTINUATION-CONDITION
         controlled-ventilation))
```

The continuation condition specifies which subplans must be completed successfully to continue with the next plan. In the subplan one-of-controlled-ventilation the CONTINUATION-CONDITION guarantees that it is only possible to start the plan weaning, when plan controlled-ventilation had been completed successfully. The alternative subplans permissive-hypercapnia or crisis-management are applied too, however the whole plan one-of-controlled-ventilation will never be completed successfully without a final successful completion of subplan controlled-ventilation.

Figure 4b is a zoom-in of Fig. 5 showing the subplan controlled-ventilation and its possible subplans using the Asbru language. In Fig. 4b two notations are used: uppercase letters followed by colons (":") indicate elements of Asbru and lowercase letters indicate particular plans, subplans, or actions. The plan controlled-ventilation is decomposed into two subplans, decreasing or increasing the ventilation setting (plan inc-ventilation and dec-ventilation) and the plan observing. The frequency of these two plans cannot be specified in advance. The number of activation periods depends on the health condition of the patient. The points (●●●) in Fig. 4b indicate these unknown repetitions. The subplan inc-ventilation is decomposed into three subplans, inc-fio2, inc-f, or inc-pip. These three subplans are nondecomposable plans (actions). Additionally, only one of these three actions can be activated at each time period, which is illustrated with the brackets. The same decomposition holds for the subplan **dec-ventilation**. The subplan **controlled-ventilation** is written in Asbru syntax:

```
(PLAN controlled-ventilation
(PREFERENCES (SELECT-METHOD BEST-FIT))
(INTENTION:INTERMEDIATE-STATE
   (MAINTAIN STATE(BG) NORMAL controlled-ventilation *))
(INTENTION:INTERMEDIATE-ACTION
   (MAINTAIN STATE(RESPIRATOR-SETTING) LOW controlled-ventilation *))
(SETUP-PRECONDITIONS
   (PIP (<= 30) I-RDS *now*)
   (BG available I-RDS
         [[_, _], [_, _],[1 MIN,_](ACTIVATED initial-phase-l#)]))
(ACTIVATED-CONDITIONS AUTOMATIC)
(ABORT-CONDITIONS ACTIVATED
   (OR (PIP (> 30) controlled-ventilation
            [[_, _], [_, _], [30 SEC, _], *self*])
       (RATE(BG) TOO-STEEP controlled-ventilation
         [[_, _], [_, _], [30 SEC,_], *self*])))
   (SAMPLING-FREQUENCY 10 SEC))
(COMPLETE-CONDITIONS
   (FiO2 (<= 50) controlled-ventilation
      [[_, _], [_, _], [180 MIN, _], *self*])
   (PIP (<= 23) controlled-ventilation
      [[_, _], [_, _], [180 MIN, _], *self*])
   (f (<= 60) controlled-ventilation
      [[_, _], [_, _], [180 MIN, _], *self*])
   (patient (NOT DYSPNEIC) controlled-ventilation
      [[_, _], [_, _], [180 MIN, ], *self*]))
   (STATE(BG) (OR NORMAL ABOVE-NORMAL)
     controlled-ventilation [[_, _], [_, _], [180 MIN,_], *self*])
   (SAMPLING-FREQUENCY 10 MIN))
(DO-ALL-SEQUENTIALLY
   (one-of-increase-decrease-ventilation)
   (observing)))
```

The intentions of subplan **controlled-ventilation** are to maintain a normal level of the blood-gas values and the lowest level of mechanical ventilation (as defined in the context of controlled ventilation therapy) during the span of time over which the subplan is executed. This subplan is activated immediately, if peak inspiratory pressure PIP ≤ 30 and the transcutaneously assessed blood-gas values are available for at least one minute after activating the last plan instance **initial-phase** (as reference point). The subplan must be aborted, if abort condition becomes true. The sampling frequency of the abort condition is 10 seconds. The subplan is completed successfully, if the complete condition becomes true. The body of the subplan **controlled-ventilation** consists of a sequential execution of the two subplans.

6 Benefits and Limitations

Applying the Asbru language to represent time-oriented skeletal plans is a very effective tool to acquire the domain knowledge needed in a structured way. The semantics for the task-specific knowledge facilitate acquisition and maintenance. Asbru places a particular emphasis on an expressive representation for time-oriented actions and world states in combination with the underlying intentions as temporal patterns to be maintained, achieved or avoided. It allows the use of different granularities and reference points to represent multiple time lines.

Asbru's representation includes the duration of actions, their success or failure, and allows time annotation of events, actions/plans, and world states with uncertainty in their appearances. Asbru has a rich set of sequential, concurrent, or cyclical operators, which enables the expression of complex procedures. Preferences, intentions, conditions, effects, and actions, are specified as various levels depending on their occurrence and evidence. The expressive representation results in an uniformly represented and organized guideline-specification library.

Nevertheless, the expressive representation of Asbru still has some limitation. In general, the medical experts were hard pressed to fill all the slots of the Asbru language and there was very little procedural, pre-compiled knowledge (protocols) found. The flexibility of the time annotation is one of the main benefits of Asbru. The ability to select different reference points is heavily used in the I-RDS protocols. Further, the ability to define different sampling intervals as shown in the above example is essential in high-frequency domains. On the one hand, it is the only way to be able to react fast in critical situations. On the other hand, it allows checking by long-term stability on a 10 minutes sampling frequency with appropriate filtering of data. In summary, the acquisition of the temporal patterns and time annotations needed is still quite difficult. In real-world high-frequency domains, the temporal dimensions are often vague or unknown.

7 Conclusion

Representing complex execution plans, such as clinical protocols, and the intentions underlying them in a sharable and acquirable form is imperative for useful, flexible automated assistance in the execution of these plans. In the manifold domains of clinical medicine and intensive care, such a task-specific representation is crucial for dissemination of modern clinical knowledge, since the use of clinical protocols will set up standards in the provision of high quality of care.

We outlined the basic concepts of an effective time-oriented representation of skeletal plans, called Asbru and proved the applicability of the Asbru syntax in the context of ventilator management in neonatal intensive care, which is based on the accurate analysis of high-frequency data.

Acknowledgements: We greatly appreciate the support given to the Austrian Research Institute of Artificial Intelligence (ÖFAI) by the Austrian Federal Ministry of Science and Transport, Vienna. Y. Shahar is supported by grants LM05708 and LM06245 from the National Library of Medicine, IRI-9528444 from the National Science Foundation. Computing resources were provided by the Stanford CAMIS project, funded under Grant No. LM05305 from the National Library of Medicine.

References

Aitken, J. S., Shadbolt, N.: Knowledge Level Planning. In Bäckström, C., Sandewall, E. (eds.), *Current Trends in AI Planning*, IOS Press, Amsterdam (1994) 33–45

Bacchus, F., Kabanza, F.: Planning for Temporally Extended Goals, *Proc. of the 13th Nat. Conference on Artificial Intelligence (AAAI-96)*, AAAI Press/The MIT Press, Menlo Park (1996) 1215–1222

Barnes, M., Barnett, G. O.: An Architecture for a Distributed Guideline Server, In Gardner, R. M. (ed.) *Proc. of the Annual Symposium on Computer Applications in Medical Care (SCAMC-95),* Hanley & Belfus (1995) 233–237

Bratman, M. E.: *Intention, Plans and Practical Reason,* Harvard Univiversity Press, Cambridge, MA (1987)

Dechter, R., Meiri, L., Pearl, J.: Temporal Constraint Networks. *Artificial Intelligence, Special Volume on Knowledge Representation,* 49(1-3) (1991) 61–95

Fox, J., Johns, N., Rahmanzadeh, A.: Protocols for Medical Procedures and Therapies: A Provisional Description of the PROforma Language and Tools, In Keravnou, E. et al. (eds.), *Proc. of the 6th Conference on Artificial Intelligence in Medicine Europe (AIME-97)*, Springer, Berlin (1997) 21–38

Friedland, P. E., Iwasaki, Y.: The Concept and Implementaion of Skeletal Plans. *Journal of Automated Reasoning,* 1(2) (1985) 161–208

Genesereth, M. R., Fikes, R. E.: *Knowledge Interchange Format, Version 3.0 Reference Manual.,* Computer Science Department, Stanford Univ., Tech.Report Logic-92-1 (1992)

Gordon, C., Johnson, P., Waite, C., Veloso, M.: Algorithm and Care Pathway: Clinical Guidelines and Healthcare Processes, In Keravnou, E. et al. (eds.), *Proc. of 6th Conference on Artificial Intelligence in Medicine Europe (AIME-97)*, Springer, Berlin (1997) 66–69

Gruber, T. R.: *Toward Principles for the Design of Ontologies Used for Knowledge Sharing, Knowledge Systems Laboratory,* Stanford Univ., TR KSL 93-04 (1993)

Guarina, N., Giaretta, P.: Ontologies and Knowledge Bases. In Mars, N. J. I. (ed.) *Towards Very Large Knowledge Base*, IOS Press, Amsterdam (1995)

Herbert, S. I., Gordon, C. J., Jackson-Smale, A., Renaud Salis, J.-L.: Protocols for Clinical Care. *Computer Methods and Programs in Biomedicine,* 48 (1995) 21–26

Hripcsak, G., Ludemann, P., Pryor, T. A., Wigertz, O. B., Clayton, P. D.: Rationale for the Arden Syntax. *Computers and Biomedical Research,* 27 (1994) 291–324

Levinson, R.: A General Programming Language for Unified Planning and Control. *Artificial Intelligence, Special Volume on Planning and Scheduling,* 76(1-2) (1995)) 319–375

Miller, P. L.: *Expert Critiquing System: Practice-Based Medical Consultation by Computer* Springer, New York, NY (1986)

Musen, M. A., Gennari, J. H., Eriksson, H., Tu, S. W., Puerta, A. R.: PROTÉGÉ-II: A Computer Support for Development of Intelligent Systems from Libraries of Components, *Proc. of the 8th World Congress on Medical Informatics (MEDINFO-95)* (1995) 766-770

Musen, M. A., Tu, S. W., Das, A. K., Shahar, Y.: EON: A Component-Based Approach to Automation of Protocol-Directed Therapy. *Journal of the American Medical Information Association,* 3(6) (1996) 367–388

Pollack, M.: The Use of Plans. *Artificial Intelligence,* 57(1) (1992) 43–68

Rit, J.-F.: Propagating Temporal Constraints for Scheduling, *Proc. of the 5th Nat. Conference on Artificial Intelligence (AAAI-86)*, Morgan Kaufmann, Los Altos (1986) 383–8

Shahar, Y., Miksch, S., Johnson, P.: A Task-Specific Ontology for Design and Execution of Time-Oriented Skeletal Plans, In Gaines, B. (ed.) *Proc. of the 1oth Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Alberta, Canada, 1996a.

Shahar, Y., Musen, M. A.: Knowledge-Based Temporal Abstraction in Clinical Domains. *Artificial Intelligence in Medicine, Special Issue Temporal Reasoning in Medicine,* 8(3) (1996b) 267–298

Sherman, E. H., Hripcsak, G., Starren, J., Jender, R. A., Clayton, P.: Using Intermediate States to Improve the Ability of the Arden Syntax to Implement Care Plans and Reuse Knowledge, In Gardner, R. M. (ed.) *Proc. of the Annual Symposium on Computer Applications in Medical Care (SCAMC-95)* Hanley & Belfus (1995) 238–242

Tate, A., Drabble, B., Kibry, R.: O-Plan2: An Open Architecture for Command, Planning, and Controll. In Zweber, M. et al. (eds.), *Intelligent Scheduling*, Morgan Kaufmann, San Francisco, (1994) 213–239

Tu, S. W., Kahn, M. G., Musen, M. A., Ferguson, J. C., Shortliffe, E. H., Fagan, L. M.: Episodic Skeletal-Plan Refinement on Temporal Data. Com. of ACM, 32(1989) 1439–1455

Uckun, S.: *Instantiating and Monitoring Skeletal Treatment Plans,* Knowledge Systems Laboratory, Stanford Univ., KSL 94-49 (1994)