Asbru, a Plan-Representing Language Modelling Time-oriented, Skeletal Plans in Sport

Silvia Miksch and Klaus Hammermüller

Vienna University of Technology, Institute of Software Technology,
Favoritenstraße 9/E188, A-1040 Vienna
<silvia,klaus>@ifs.tuwien.ac.at

Abstract. Workout planning is a very time-consuming task, where lots of interactions between different exercises have to be observed. The available knowledge is partly very large but incomplete and cannot automatically be transfer to the current situation of an individual athlete. There are different problems in this area: the transfer of available knowledge, individual adaptation, and effective evaluation of intended effects after the planned exercises were performed.
Skeletal plans are a very powerful way to tackle these problems. These plans are as well human readable and support automated processing of the gathered knowledge. It is a huge step from "free text" representation to a notation where a computer may help a coach effectively. We try to split this big step into some smaller steps by looking at the specific tasks which are performed in workout planning and evaluation. These tasks are supported by a set of "problem solving methods" which can be implemented independently based on a common plan representation we called *"Asbru"*.

1 Introduction

To support workout planning; sport-specific concepts, plans for homogeneous groups of athletes and example plans are established in many areas of sport; following we call them "guidelines". Generally speaking, they are represented as free texts, tables, or diagrams. These documents are very important to transfer knowledge between different coaches, they tell the athlete what to do and enable a supervised development of a team; but these documents are also far from perfect, because they do not integrate the input from different sources to a consistent workflow participating different actors. Moreover, they do not allow automatical support for verification or quality assessment.

Currently a guideline is available in some text-version. However in practice planning work maybe done using a spreadsheet, which is used to handle the real-work data produced by the athlete's performance. Neither the planning world itself nor the connection to the execution of these plans are supported. These problems are very familiar to planning domain. In the Asgaard/Asbru[1] project [1, 8] a number of methods are being developed to deal with the problems of clinical therapy planning in medicine. During this project we have explored many analogies to other time-oriented planning domains like workout planning in sports, and we try to transfer these domain-specific solutions to a more general concept.

Looking at the planning process itself, we have three major phases outlined in Figure 1:

- Planning the actions based on future projections of the intended effects, which are related to the goal of the plan;
- Executing the plan in reality and maybe adapting the plan supporting the high-level goals of the workout performance;
- Analyzing the performance by compared intended effects and realized effects by the processed exercises.

[1] In Norse mythology, *Asbru* (or *Bifrost*) was the bridge from our world to *Asgaard*, the home of the gods.

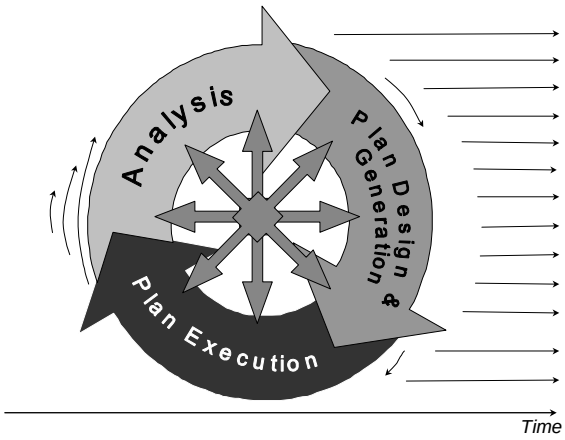

Fig. 1. Intertwinedness of the Plan Process

This process is continuously iterating and evolves over time. A better model of this might be a screw which would improve during circulating round its center; we call this "intertwinedness". There are three clearly distinguishable phases: planning, executing and analysis, that are connected very strongly with each-other. To support one phase effectively an efficient information-transfer from the other phases has to be established. This is the major gap in the actual state of the planning work: We have no free and compatible flow of information to perform the tasks at this process effectively.

To make our methods supporting these tasks usable for the actor (like a coach) there is the need for an intuitive visualization, discussed in [5]. This part of the *Asgaard* project is named *"AsbruView"* and will be part of the example in section 4. (We don't think that a coach should handle *Asbru* in its "native" form which we present in this paper.) *Asbru* itself is a language used for representing plans in a LISP-like syntax which is outlined in section 2. The description about the problem specific task support with *Asbru* is added in section 3. An overview about related work on different kinds of plan-representations is given in section 5. Future aspects and drawbacks are discussed in conclusion in section 6.

2 Concepts of *Asbru*

The Key features of *Asbru* are

- Hierarchical decomposition of Plans, that means we can start with a very simple plan, saying e.g. "we want the athlete to run faster" and adding refining sub-plans to this high-level plan;
- A powerful concept of temporal annotations, which allow handling complex dimensions and interactions of different time-lines;
- Every plan is structured into different components: Preferences, Intentions, Conditions, Effects and Plan Layout, which model specific important parts of the collected knowledge like the "intended" goal of the plan.

2.1 Time-Oriented, Skeletal Plans

The various elements of a skeletal plan is outlined in Figure 2. On the one hand a plan consists of different components: Preferences, Intentions, Conditions and Effects. An Intention for example, describes the goal which should be maintained or avoided. On the other hand, every plan can be decomposed into sub-plans or atomic actions. An atomic action is a plan which does not hold any sub-plans anymore.

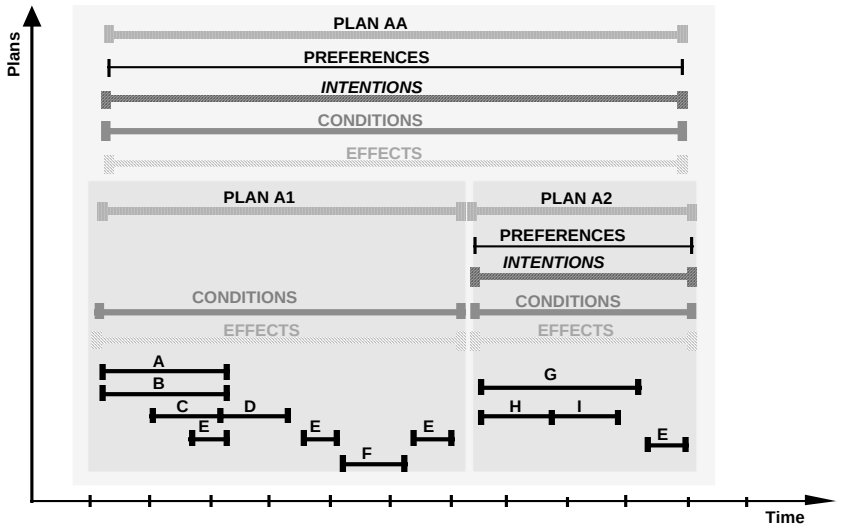

Fig. 2. Outline of a Hierarchical Skeletal Plan

The idea behind the "skeletal" property of the plan is to make some general guidelines in order to deal with a specific problem, such as the workout-plan for an athlete after some kind of illness or a general plan in a regular training process. The point is, these plans are *not* individual plans but reusable concepts which orient on certain situations or conditions of the athletes. Once such a plan is selected for an particular athlete it may be adapted in order to fit to the individual needs according to the current conditions.

In Figure 3 the general idea of the use of skeletal plans is described. Assuming there are two different plans, A and B, we are looking at the real-world states represented in the available real data; and we are selecting the plan which "fits" better to the actual situation. As the *Asgaard* framework is an open system integrating the actor, in this point we try to give some decision-support for selecting a plan.

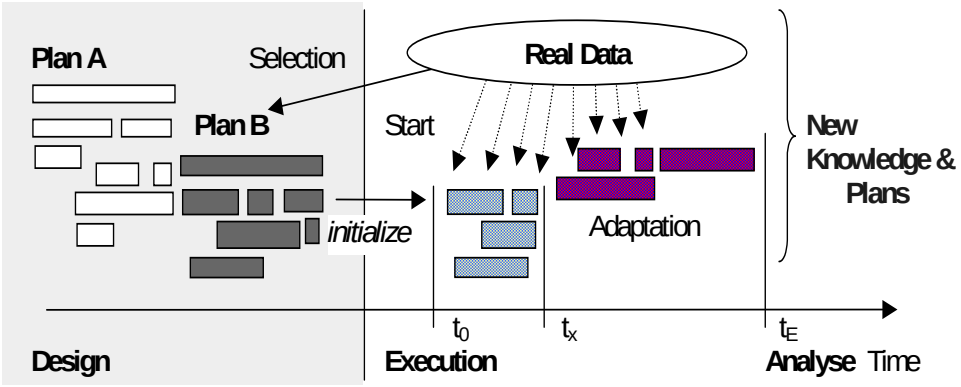

Fig. 3. Using Skeletal Plans

The next step is to initialize the plan (t0) with the available data. A plan may consist of different sub-plans; starting to execute the selected plans, we will start with some subset of sub-plans and actions performing workout exercises.

During the execution of such a workout plan the real-world state may change, new data about the condition of the athlete are available and the chosen plan is "filled" with this information. During this process the different components of a plan are monitored, e.g. if the intended aim of the workout plan stays achievable or not.

If at some point in time (tx) the selected plan doesn't fit to the real-world process any more, an other plan has to take place to achieve the intended aims. What happened here is that e.g. an illness of the athlete makes it impossible to continue with executing the plan and the *Asgaard* framework supports this change in the "real-world-environment".

After the plans are completed (te) a retrospective analysis of the history is possible. The plans with their intentions, the criteria for selecting a workout-plan, the processed exercises as well as information about the athletes results are available in a homogeneous representation. The retrospective analysis may result in revising existing workout plans or adding new workout plans.

2.2 Time Annotations and Temporal Pattern

As human being we all handle time very intuitively. For computer and the power of semantic languages like *Asbru* the notation of time is a crucial point. Using time-stamps is the most simplest but also the least powerful way solving this problem.

Describing actions in sports we normally do not know *exactly* how long e.g. regeneration takes place and when there is the optimal time for the next exercise in a individual cases. However, we have to handle this problem to produce useful statements and therefore we have a very powerful notation we outlined in Figure 4.

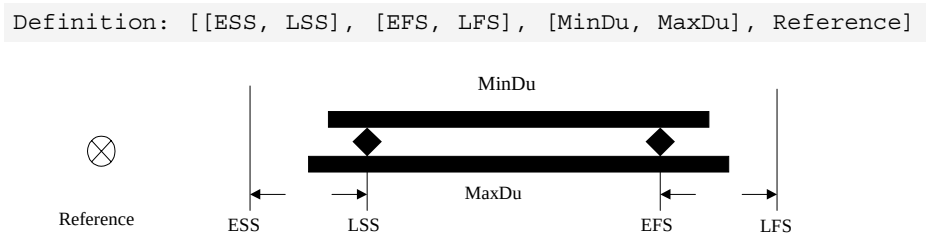

Fig. 4. Notation of Time Annotations

A time-annotation is a set of relative time-shifts to a specified *reference* point, describing earliest (*ESS*) and latest starting shift (*LSS*) as well as earliest (*EFS*) and latest finishing shift (*LFS*) and a minimum (*MinDu*) and maximum duration (*MaxDu*) of a plan.

For Example: "starts 24 to 30 hours after the last exercise, ends 25 to 32 hours after the last exercise, and lasts 1 to 2 hours." would look like: `[[24 HOURS, 30 HOURS], [25 HOURS, 32HOURS], [1 HOURS , 2 HOURS], LAST-EXERCISE]`.

The definition does not necessarily need to be complete specified, the framework will handle incomplete time-definitions as far as possible. Time-annotations are the backbone of *Asbru*, talking about "values"; we like to connect these "values" with its abstraction [6] (e.g. in a defined context a pulse may be "high", not "140") and a representation of time. We call this *"Temporal Pattern"*, which consists of:

1. A *parameter proposition*: a parameter (or its abstraction), context, and time annotation or
2. a *combination* of multiple parameter propositions or
3. a *plan-state* associated to an instantiated plan (plan pointer) and a time annotation.

A simple example: for a special kind of interval-workout the pulse should be low for 5 to 7 minutes before starting the next interval:

```
(STATE(pulse) LOW WORKOUT [[0 MIN, 0 MIN], [_, _],[5 MIN, 7 MIN], INTERVAL])
```

2.3 Meaning of "Intention-Based"

Going more into details of *Asbru*, we have a look at the "components" of this language. We will discuss here only one component, the "Intentions". A more complete description of all components is published in [4].

Definition: Intentions are temporal patterns of actions or states, to be maintained, achieved, or avoided.

- *Intermediate state:* The state(s) that should be maintained, achieved, or avoided during the applicability of the plan;
- *Intermediate action:* The action(s) that should take place during the execution of the plan;
- *Overall state pattern:* The overall pattern of states that should hold after finishing the plan;
- *Overall action pattern:* The overall pattern of actions that should hold after finishing the plan.

For example: "The speed of an athlete shall be increased by 0,5% in a period of time between 5 and 7 weeks." This information itself makes a valid plan in our representation language *Asbru*. A more detailed plan may define: "You have to increase the extent of your workout by 4%". Maybe the athlete has not this additional amount of time but may achieve the goal by adding some specific strength exercises. If this modification of the plan fulfill the intention, the plan has been processed successfully.

As we defined before, a plan consists of a set of different components, like this intentions. Not every plan or sub-plan has to define all of these components. As *Asbru* is a hierarchical language, missing components are propagated through this hierarchical structure. Maybe even an expert will not be able to define all the components of the plan-details, but in the end the question "Is the athlete faster?" will be answerable. Of course, the more detailed the definition of the different components are, the more precise and helpful can the *Asgaard* framework support the actors.

3 The *Asgaard* Framework: Tasks Supported by *Asbru*

3.1 Task-specific Problem-Solving Methods

We think about planning as an open loop, integrating the actor into the process. Some planning tools in computer science have a different point of view: They try to replace the human actor in specific tasks which are performed; we have an other strategy. The *Asgaard* framework tries to support the actor by the performance of the different tasks, and wants to help to focus on the main purpose of the work.

As you can see in Figure 5 the computer shall support the user applying *task-specific problem-solving methods*, *"PSM"* using *workout concepts* or guidelines which are acquired at "design time" and *time-oriented input data* from the athletes workout at "execution time". Additionally, these information are collected in a *sharable plan-specification library* to enable the re-use of collected knowledge.

The output of these *PSM* is a set of processed procedures like *plan-monitoring* (is the execution of the plan on track in reaching the intended aims), *Critiquing* and other high level information. As outlined before, we distinguish between design-time, when the future actions are planned and execution-time when planned actions are performed in real world [3].

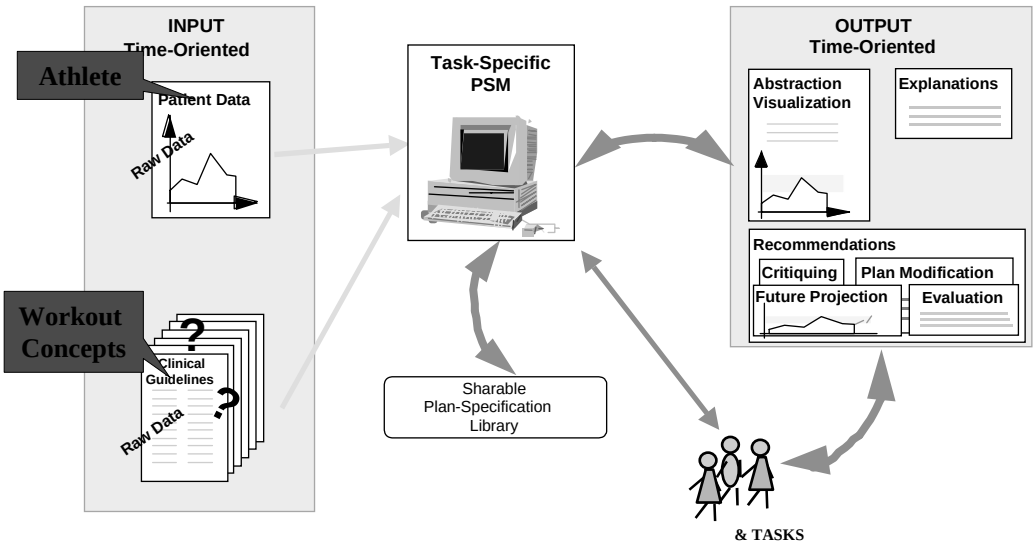

Fig. 5. The *Asgaard / Asbru* Concept

3.2 Methods Supported at Design Time

- *Plan Generation:* fits suitable skeletal plans together;
- *Advanced Plan Editing:* is an editor guided by the knowledge-roles to give full access to expert users, who tune features of the knowledge roles;
- *Domain-Specific Annotations:* supplies structured support to write domain assumptions or domain functions;
- *Plan Verification:* examines the correctness of interrelated skeletal plans by a three-level detection of anomalies (method semantics);
- *Plan Validation:* compares the intended states against the prescribed actions and intended plans (domain semantics);
- *Plan-Scenario Testing:* applies scenarios of protocols to test their functionalities and their course of activities; it is partly included in the plan visualization;
- *Plan Visualization:* is a graphical editor to design and to browse the topological and the temporal views of the connected plans.

3.3 Methods Supported at Execution Time

- *Plan Selection:* chooses applicable skeletal plans from the plan library according to the athletes's state, the plan's overall intentions, and plan effects;
- *Plan Adaptation:* adjusts the parameters of a skeletal plan according to the athletes's state and the medical environment;
- *Plans Execution:* performs according to the execution-plan's prescribed actions;
- *Plans Monitoring:* oversees and administers whether the executed plans are still applicable at the particular time of execution at the sampling frequencies given in the temporal patterns; executed plans can be suspended and reactivated;
- *Plan Modification / Alternatives:* chooses alternative actions or plans, which are relevant at this time for achieving a given intention
- *Plan Critiquing:*
  1. 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;
  2. 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?

- *Plan Evaluation:* examines retrospectively, if the executed skeletal plans achieved the desired effects according to the patient's state, intentions, executed actions, and plan effects;
- *Executed Plan Visualization:* visualizes, which plans have been executed (when and how);
- *Plan Rationale / History:* bookkeeping of events, states, intentions and performed actions; generates on-line help information and explanations about plan selection, adaptation, execution states, success and failure of plans integrating the domain-specific annotations.

4 Example

Figure 6 shows an medical example how a user face the Asbru language. We call this tool *"AsbruView"* [5] and it shall help to design the general outline of a plan.

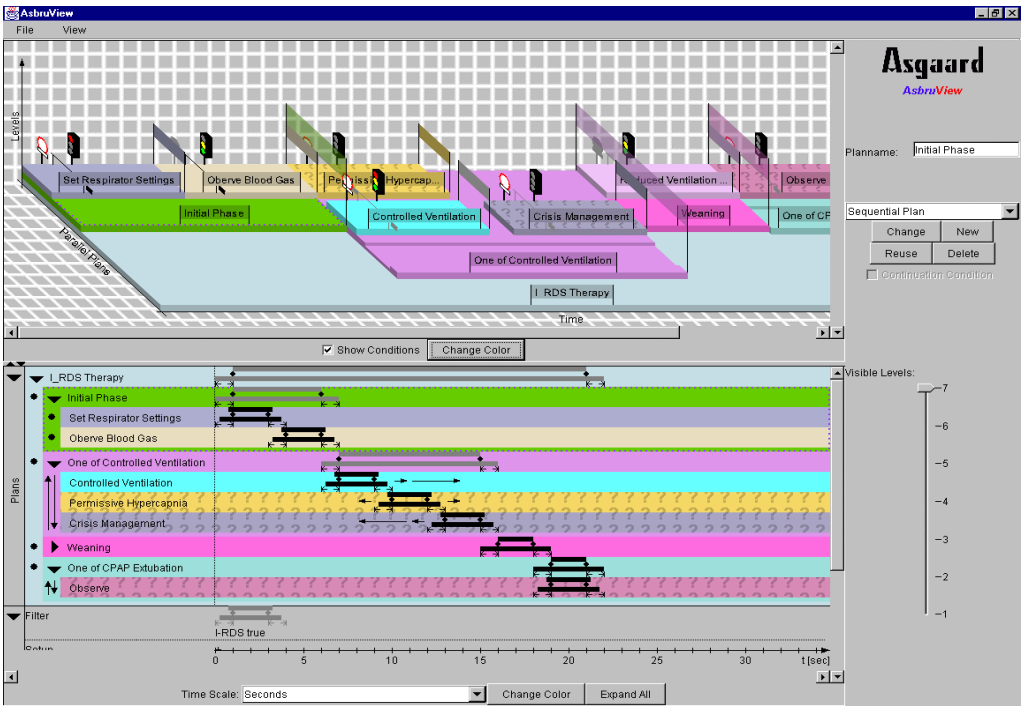

Fig. 6. Screenshot of the *Asbru*View Prototype

What you see here in the upper-half of the screenshot is the hierarchical decomposition of a plan in its sub-plans. We are using metaphor graphics to give an easy access to the huge amount of details of such a plan. Each one of these "running tracks" is a sub-plan which has to be processed at execution-time. The traffic signs represent the conditions of the plan, a special component of *Asbru*. The tracks are ordered by different means, visualizing that some things have to happen sequentially, parallel or any other order.

On the lower half, the temporal dimension (see section 2.2) of the plans are decomposed to enable an exact manipulation of the time-dimensions and its uncertainties. We also work intensively with colors and textures to annotate the additional dimension: question-marks identify optional sub-plans and the grayed time-annotations represent uncertain definitions.

5 Related Work

During the past 15 years, there have been several efforts to create automatic reactive planners to support the process of protocol-based care over significant periods of time. In the prescriptive

approach, an active interpretation of the guidelines is given; examples include *ONCOCIN* [9] in the oncology domain, *T-HELPER* [7] in the AIDS domain, and *DILEMMA* [2], and the European *PRESTIGE Health-Telematics* project, as general architecture.

In the critic approach, the program criticizes 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. A task-specific architecture implementing the criticizing process has been generalized in the *HyperCritic system* [10]. Task-specific architecture assign well-defined problem-solving roles to domain knowledge and facilitate acquisition and maintenance of that knowledge.

None of the current guideline-based systems have a sharable representation of guidelines that has knowledge roles specific to the guideline-based task, is machine and human readable, and allows data stored in an electronic record to invoke an application that directly executes the guidelines logic and related tasks, such as critiquing. Such a machinereadable language and the task-specific problem solving methods need to solve that problem described in [8]. However, a much more detailed comparison between different planning concepts is given in [3].

6 Conclusion

There are very familiar tasks to be done in the medical domain as well as in the world of sports or other time-oriented planning domains, which we demonstrated in the list of tasks in section 3.

Thinking about *Asbru* itself, we try to model *patterns* (see Intentions at section 2.3) and work with qualitative abstractions of values [6] to make knowledge portable between different athletes. We also handle the complex task of time with a very powerful representation (see section 2.2).

Asbru's Drawbacks:

- Acquisition of the temporal patterns and time annotations needed is still difficult, temporal dimensions are often vague or unknown;
- Troublesome to cope with all possible orders of plan execution and all the exception conditions that might arise. Coaches use a lot of background knowledge, which is hard to acquire and to represent as a skeletal plan.

Asbru's Benefits

- Transfer of knowledge between coaches as well as between coach and athletes;
- Verification and validation of the plan itself;
- Evaluation of workout plans and the possibility to produce new knowledge retrospectively;
- Support of different knowledge roles;
- The possibility to describe and process complex temporal dimensions;
- Reuse of existing knowledge and acquired plans;
- To establish quality assurance in workout practice.

In conclusion we think the use of a time-oriented descriptive planning language like *Asbru* can fill the gap between future plan projections and flexible executions of these plans in order to perform efficient workout planning and execution. Automated support may help coaches to focus on their primary work: treating athletes and enabling them a high-level knowledge-transfer as well as quality assurances during the process of workout.

Acknowledgements

We wish to thank Robert Kosara, Katharina Renath and Andreas Seyfang for valuable suggestions and discussions and for the chance to work with them. This project is supported by *"Fonds zur Förderung der wissenschaftlichen Forschung" (Austrian Science Foundation)*, P12797-INF.

References

1. The Asgaard-project: http://www.ifs.tuwien.ac.at/asgaard/.
2. S. I. Herbert, C. J. Gordon, A. Jackson-Smale, and J.-L Renaud Salis. Protocols for clinical care. *Computer Methods and Programs in Biomedicine*, 48:21–6, 1995.
3. S. Miksch. Plan Management in the Medical Domain. *AI Communications*, 4, 1999.
4. S. Miksch and R. Kosara. Communicating Time-Oriented, Skeletal Plans to Domain Experts Lucidly. In *10th International Conference of Database and Expert Systems Applications (DEXA99)*, 1999.
5. S. Miksch, R. Kosara, Y. Shahar, and P. Johnson. AsbruView: Visualization of time-oriented, skeletal plans. In *Proceedings of the 4th International Conference on Artificial Intelligence Planning Systems 1998 (AIPS-98)*. Carnegie Mellon University, AAAI Press, June 7–10 1998.
6. S. Miksch, A. Seyfang, W. Horn, and Popow C. Abstracting Steady Qualitative Descriptions over Time from Noisy, High-Frequency Data. In *Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making, AIMDM'99*, Berlin, 1999. Springer. in print.
7. M. A. Musen, C. W. Carlson, L. M. Fagan, S. C. Deresinski, and E. H. Shortliffe. T-helper: Automated support for community-based clinical research. *Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care (SCAMC-92)*, pages 719–23, 1992.
8. Y. Shahar, S. Miksch, and P. Johnson. The Asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines. *Artificial Intelligence in Medicine*, 14:29–51, 1998.
9. S. W. Tu, M. G. Kahn, M. A. Musen, J. C. Ferguson, E. H. Shortliffe, and L. M. Fagan. Skeletal-plan refinement on temporal data. *Communications of the ACM*, 32:1439–55, 1989.
10. J. Van der Lei and M. A. Musen. A model for critiquing based on automated medical records. *Computers and Biomedical Research*, 24(4):344–78, 1991.