CareVis: Interactive Visualization Methods to Support Protocol-Based Care

Wolfgang Aigner
Silvia Miksch

Vienna University of Technology
Institute of Software Technology & Interactive Systems (ISIS)

Asgaard-TR-2005-8 July 2005

Authors: **Wolfgang Aigner**
Silvia Miksch

{aigner, silvia}@ifs.tuwien.ac.at
http://ieg.ifs.tuwien.ac.at

Contact: **Vienna University of Technology**
Institute of Software Technology & Interactive Systems (ISIS)

Favoritenstraße 9-11/188
A-1040 Vienna
Austria, Europe

Telephone: +43 1 58801 18833
Telefax: +43 1 58801 18899
Web http://ieg.ifs.tuwien.ac.at

CareVis: Interactive Visualization Methods to Support Protocol-Based Care

Wolfgang Aigner and Silvia Miksch

Institute of Software Technology & Interactive Systems (ISIS)
Vienna University of Technology
Favoritenstraße 9-11/188, A-1040 Vienna, Austria, Europe
Tel: +43 1 58801 18833, Fax: +43 1 58801 18899

Abstract

Objective: Computer supported protocol-based care aims to aid physicians in the treatment process. The main focus of current research is directed towards the formal methods and representations used "behind the scenes" of such systems. This work is trying to build the bridge towards medical domain experts in order to support their daily work tasks of getting aquanited with treatment methods, treating patients, and analyzing data.

Methods and Material: We describe the development of *CareVis* – interactive visualization methods to support protocol-based care. The user-centered development approach applied for these interactive visualization methods has been guided by user input gathered via a user study, design reviews, and prototype evaluations.

Results: CareVis integrates and combines classical data visualization with the visualization of treatment information in terms of logical structure and temporal aspects. We provide multiple simultaneous views to cover different aspects of a complex underlying data structure of treatment plans and patient data. The tightly coupled views use visualization methods well-known to domain experts and are designed to facilitate users' tasks.

Conclusion: Our 3-step evaluation process showed that the integrated visualization of medical treatment plans and patient data helps to ease the complex and demanding tasks physicians have to face daily.

Key words: protocol-based care, information visualization, clinical guidelines, treatment plans, patient data, user-centered design

Email address: {aigner, silvia}@asgaard.tuwien.ac.at (Wolfgang Aigner and Silvia Miksch).

URL: http://www.asgaard.tuwien.ac.at (Wolfgang Aigner and Silvia Miksch).

1 Introduction

Computer supported protocol-based care is a field of research that aims for semiautomatically supporting the treatment process along protocols [1] by the use of information technology. The core entity, medical treatment plans, are complex documents, currently mostly in the form of prose text including tables and figures [1]. Protocol-based care utilizes clinical protocols to assist in quality improvement and reduce process irregularities. Such clinical protocols are a standard set of tasks that define precisely, how different classes of patients should be managed or treated. They can be seen as reusable components of a particular care process. Treatment planning covers the whole process of selecting and executing a particular clinical protocol for a specific patient.

Currently, most of the data is organized in paper-based records including general patient data, treatment steps, lab results, medications, and much more, making it hard to get a comprehensive overview or relate data of different kinds to each other. Several research projects are dealing with the formalization of this kind of documents in order to facilitate computer based execution support (see [2] for an overview). Hereby, knowledge acquisition, formalizing unstructured treatment documents, creating domain models, data abstraction, executing plans (semi-) automatically, and the like are the major concerns of research. Not much work has been done in order to communicate the computerized treatment plans to the medical staff and even less for combining this with the presentation of patient data when treating a patient along a plan for monitoring and analytic tasks. The integrated visualization of medical treatment plans and patient data could be of great assistance to ease the complex and demanding tasks physicians have to face daily. As important as the task of feeding real world information into a computer system in a structured and meaningful way and processing it, is presenting and communicating this information to human domain experts, in our case physicians, nursing-, and other medical personnel. This presentation and communication has to be done in a clear, simple, and comprehensible way, preferably familiar to the end users in order to keep the learning effort as low as possible.

Now, we will give a short introduction of the plan representation language *Asbru* followed by a task and data analysis of the problem domain in Section 3. After the discussion of related work we will describe the first step of our user centered development approach, the acquirement of physicians' needs, in Section 5. Subsequently, our visualization approach *CareVis* will be presented along with evaluation and prototype implementation issues in Sections 6 and 7. Finally, we describe how users' tasks are supported and sum up our findings.

[1] Throughout this article, the expressions *clinical guideline*, *guideline*, *treatment plan*, *protocol*, and *plan* will be used interchangeably.

2 The Plan Representation Language Asbru

Asbru is a time-oriented, intention-based, skeletal plan-specification representation language that is used in the *Asgaard* Project [2] to represent clinical guidelines and protocols. *Asbru* can be used to express clinical protocols as skeletal plans [3] that can be instantiated for every patient (for an example see Fig. 1). It was designed specific for the set of plan-management tasks [4]. *Asbru* enables the designer to represent both the prescribed actions of a skeletal plan and the knowledge roles required by the various problem-solving methods for performing the intertwined supporting subtasks. The major features of *Asbru* are that;

- prescribed actions and states can be continuous;
- intentions, conditions, and world states are temporal patterns [3];
- uncertainty in both temporal scopes and parameters can be flexibly expressed by bounding intervals;
- plans might be executed in sequence, all or some plans in parallel, all or some plans in a particular order or unordered, or periodically;
- particular conditions are defined to monitor the plan's execution; and
- explicit intentions and preferences can be stated for each plan separately.

2.1 Example

Figure 1 shows parts of an *Asbru* plan for artificial ventilation of newborn infants. The guideline is represented in XML and contains domain definitions and a set of plans. The *ventilation plan* consists of conditions and the plan body including a sequential execution of the *initial plan* and *controlled ventilation plan*.

Basically, an *Asbru* plan can be seen as a template. This template gets instantiated whenever the plan gets executed. Additionally, more than one instance might be created for a single plan. This pattern can be seen as an analogy to the Class-Object relationship in Object-Oriented Programming.

Since a plan is represented in XML, it is basically readable to humans. But understanding a plan in such a representation needs a lot of training as well

[2] In Norse mythology, *Asgaard* was the home of the gods. It was located in the heavens and was accessible only over the rainbow bridge, called *Asbru* (or *Bifrost*) (For more information about the *Asgaard* project see `http://www.asgaard.tuwien.ac.at`).

[3] The temporal patterns used in *Asbru* are explained in more detail in Section 6.1.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plan-library SYSTEM "asbru_7_3.dtd">
<plan-library>
    <domain-defs>
        <domain name="controlled_ventilation_domain">
            ...
        </domain>
    </domain-defs>
    <plans>
        <plan-group>
            <plan name="ventilation_plan">
                <intentions> ...  </intentions>
                <conditions>
                    <complete-condition>
                        <constraint-combination type="and">
                            <parameter-proposition parameter-name="FiO2">
                                <value-description type="less-or-equal">
                                    <numerical-constant value="40"/>
                                </value-description>
                            ...
                        </constraint-combination>
                    </complete-condition>
                    <abort-condition>
                        <constraint-combination type="or">
                            <parameter-proposition parameter-name="FiO2">
                                <value-description type="greater-than">
                                    <numerical-constant value="90"/>
                                </value-description>
                            ...
                        </constraint-combination>
                    </abort-condition>
                </conditions>
                <plan-body>
                    <subplans type="sequentially">
                        ...
                        <plan-activation>
                            <plan-schema name="initial_plan"/>
                        </plan-activation>
                        <plan-activation>
                            <plan-schema name="controlled_ventilation_plan"/>
                        </plan-activation>
                    </subplans>
                </plan-body>
            </plan>
            ...
            <plan name="controlled_ventilation_plan">
                <plan-body>
                    <subplans type="parallel">
                        ...
                        <plan-activation>
                            <plan-schema name="handle_PCO2_plan"/>
                        </plan-activation>
                        <plan-activation>
                            <plan-schema name="handle_tcSaO2_low_plan"/>
                        </plan-activation>
                        <plan-activation>
                            <plan-schema name="handle_tcSaO2_high_plan"/>
                        </plan-activation>
                    </subplans>
                </plan-body>
            </plan>
            ...
        </plan-group>
    </plans>
</plan-library>
```

Fig. 1. An example of Asbru 7.3 code: Parts of a clinical treatment plan for artificial ventilation of newborn infants.

as semantic and syntactic knowledge about the representation language. It is cumbersome, and surely not suited for physicians. Therefore, the formal representation needs to be translated into a form familiar to domain experts to be able to communicate the logic and temporal aspects of a computerized treatment plan.

3 Task and Data Analysis

3.1 User Tasks and Scenarios

To illustrate the different tasks of medical personnel, we created three use scenarios [5] of physicians in protocol-based care.

3.1.1 Scenario 1

Markus Zolte, assistant doctor in training in internal medicine, will be working in the pediatrics department for the next few months and is exploring various treatment methods for new born infants. He informs himself about hyperbilirubinemia by walking through the related treatment protocol. He is interested in the logical workflow and explores the treatment plan. After the first walkthrough of the hyperbilirubinemia protocol, Markus Zolte goes back to the intensive phototherapy part and wants to know in which cases this plan is stopped. He is also interested which part of the complete treatment plan he is viewing right now. Furthermore, he wants to see all other parameters and variables that are getting used in this treatment plan.

3.1.2 Scenario 2

Andrea Habacher, assistent medical director of internal medicine, just completed the treatment of a patient using the controlled ventilation plan. Now, she wants to analyze different parts of the treatment along with measured patient data. She starts by examining how long different phases of the plan took in relation to others. The "handle PCO2 plan" is of particular interest to her. She also wants to see the PCO2 value for examining relations between plan execution and PCO2 values. Because there is a significant discontinuity of the PCO2 value within this plan, she recalls the sub-steps taken in the "handle PCO2 plan". Furthermore, she wants to see when the particular steps were conducted. After that, she is interested in if and how the PCO2 values influenced the "patient-state" parameter.

3.1.3 Scenario 3

Heinrich Kovanic, assistant medical doctor in an intensive care unit (ICU), is currently treating a patient who suffers from hyperbilirubinemia. He examines the "TSB" (total serum bilirubin) and "TSB-change" values and wants to review the patient record for getting basic patient information. After that, he investigates all incoming parameters and encounters a rapid increase of the TSB value that happened two hours ago. He wants to find out which plan or action took place at that time. Furthermore, he examines the parameter constraints defined by the plan conditions. After encountering the reason for the value change, he wants to go back to the current position of plan execution.

3.1.4 Tasks

Summarizing the essentials of these scenarios, three fundamental user tasks can be identified:

- Becoming acquainted with a specific treatment method and observed patient's parameters.
- Guidance in the treatment process (run-time support while treating a patient via monitoring patient status, presenting upcoming treatment steps, and providing a treatment history).
- Analyzing the treatment process (observed data together with treatments).

3.2 Data Characteristics

The underlying data for the tasks identified above can be broken down in three categories:

- treatment plan specification data
- treatment plan execution data (instantiation and execution of a plan)
- patient data (time oriented)

Analyzing the type and structure of this data formulated in *Asbru* yields a number of visualization relevant characteristics:

- time-oriented data (execution data and planning data including a rich set of time attributes to represent uncertainties)
- logical sequences
- hierarchical decomposition
- flexible execution order (sequential, parallel, unordered, any-order)
- non-uniform element types
- state characteristics of conditions

Starting from this basis of user tasks and data as well as visualization relevant characteristics, we examined related work as highlighted in the upcoming section.

4 Related Work

We investigated related work in the areas of medical treatment planning, information visualization, and commercial medical software as described in the following.

4.1 Medical Treatment Planning

4.1.1 Clinical Algorithm Maps

The most widely used visual representation of clinical guidelines are so-called *flow-chart algorithms*, also known as *clinical algorithm maps* [6]. A standard for this kind of flow-chart representation has been proposed by the *Committee on Standardization of Clinical Algorithms* of the *Society for Medical Decision Making*:

"However, since algorithmic logic is wired implicitly into a protocol, it is difficult to learn an algorithm from a protocol. By contrast, flow-chart algorithms, or clinical algorithm maps, are uniquely suited for explicitly communicating conditional logic and have therefore become the main format for representing a clinical algorithm clearly and succinctly." [7]. The proposed standard includes a small number of different symbols and some rules on how to use them. One additional feature to standard *flow-charts* are *annotations* that include further details, i.e. citations to supporting literature, or clarifications for the rationale of decisions.

A big advantage of using flow-charts is that they are well known among physicians and require minimal additional learning effort. A drawback of basic flow-chart representations is their immense space consumption if more complex situations are depicted where overview is lost easily. Temporal information can only be represented implicitly on a very coarse level in terms of an item's relative position within a sequence. Furthermore, flow-charts cannot be used to represent concurrent tasks or the complex conditions as used in *Asbru*. Clinical algorithm maps were intended to be used on paper and have never been enriched by computer support such as navigation or versatile annotation possibilities.

4.1.2 AsbruView

AsbruView [8,9] is a graphical tool that supports authoring and manipulation of *Asbru* plans. AsbruView utilizes metaphors of running tracks and traffic control to communicate important concepts and uses glyphs to depict the complex time annotations used in *Asbru*. The interface consists basically of two major parts, respectively views – one captures the topology of plans, whereas the second one shows the temporal dimension of plans but no depiction of plan and patient data is possible. The intention of AsbruView is to support plan creation and manipulation. It is not supposed to communicate the combination of logic, structure, and temporal aspects of an *Asbru* plan and patient data during execution or analysis.

4.1.3 Other Scientific Projects

Other scientific work [10–12] on visual representations focused on visualizing patient data over time or plan execution over time. Research projects dealing with protocol-based care include *GLARE*, *GUIDE*, *Protégé*, *GLIF*, *PROforma*, and *GASTON*. (A comprehensive overview of related protocol-based care projects can be found in [2] and [13].)

Only some of the available projects dealing with protocol-based care provide any graphical representations. The listed ones include such graphical representations, but most of them only focus on authoring plans. They use a flowchart- or workflow-like presentation depicting the elements used in their formal representation. A more detailed discussion of the quoted projects can be found in [14].

These tools make authoring clinical protocols easier especially for non-computer-scientists, but authoring clinical guidelines and communicating complete protocols to domain experts are two rather different tasks with different goals. Additionally, the presented techniques use graphical representations which are not familiar to domain experts and mix state and flow-chart characteristics within a single diagram. Understanding such representations and using them for plan authoring requires a considerable amount of learning effort.

In contrast to that, our goal is the intuitive communication of logical in conjunction with temporal aspects of a treatment plan and patient state parameters. Whereas the presentation of and navigation within guidelines is paramount along with offering easy access to linked information and in-depth explanations.

4.2 Information Visualization Methods

4.2.1 Visualizing Logical Sequences

Other possibilities to visualize logical sequences away from flow-charts are *Structograms (Nassi-Shneiderman Diagrams)*, *PERT charts*, *Petri nets*, and *State Transition Diagrams*. These techniques focus on other purposes and some of them are more powerful and expressive than flow-charts. But none of them offers a notion for depicting hierarchical decomposition, flexible execution order, and the state characteristic of conditions together in their basic forms as needed for representing *Asbru* plans.

4.2.2 Visualizing Hierarchical Data

The most popular techniques for visualizing hierarchical data are *Trees*. Further techniques for that matter are *Treemaps* [15] that introduce an additional dimension by proportional space assignment. But these 2D techniques have no notion to depict logical sequences, concurrency, or states.

4.2.3 Visualizing Time-Oriented Data

Time is a very important data characteristic but methods for visualizing time other than in time-series plots are not well known. The probably best known method among them are *GANTT charts* and their utilized *Time Lines*. An extension of *Time Lines* are *LifeLines* [16,12] that have been used for example to visualize personal histories. A drawback of these methods is that they mostly work retrospectively, thus only depict temporal attributes in the past. To overcome this limitation, other visualization techniques like *Temporal Objects* [17], *Paint Strips* [18], and *SOPOs* [19,20] were developed. These techniques can be used to visualize complex notions of time like temporal uncertainties that can be utilized to depict future planning data. The main flaw of the presented techniques is that, except *GANTT charts*, they cannot depict hierarchies and logical sequences can only be represented implicitly.

4.3 Commercial Medical Software

A very high portion of the offered commercial software products in medicine deal with administrative issues such as patient data management or billing. Only very few include any visualization parts and even less offer functionality to aid treatment planning.

We examined a number of non-administrative software products that use graphical representations in general (not only focused on protocol-based care), for the reason of compiling a set of graphical representations most commonly used and that are familiar to most physicians [14].

All of the examined products are rather data-centric and the most popular form of data representation is using tables where numerical and textual data is organized in spreadsheets. None of the investigated products offered a way of visualizing treatment planning logic at all.

We think that besides examining related work on a scientific basis and investigating commercial products it is absolutely necessary to involve end-users from the very beginning. Only this can ensure the incorporation of the users' valuable experience, knowledge, and desires, thus increasing quality and acceptance dramatically. This user-centric development was started by carrying out a user study as described in the following section.

5 User Study to Acquire Physicians' Needs

A step of major importance for requirement analysis in our development process was to conduct a user study with eight physicians to gain deeper insights into the medical domain, work practices, application of guidelines in daily work, users' needs, expectations, and imaginations.

Most of the interviewed physicians work at different departments for critically ill patients at the General Hospital of Vienna (AKH Wien). The AKH Wien is a university clinic which means that employed physicians' work also includes scientific research. Conducting an interview took on average about 45 minutes and led to interesting, but not too surprising results and insights. (Detailed results and interview guidelines can be found in [14].)

Fundamental issues for the interviewed physicians were rather practical ones. Most importantly the system has to save time – no one would use a system if it would take more time as working without it. Another major issue is that learning effort for using the system has to be minimal. The system should be intuitive, simple, and clearly structured without complex menu structures or functions.

It became apparent that clinical guidelines are generally depicted by a special form of flow-charts named *clinical algorithm maps* as proposed in [7] and are widely known. GANTT charts were known among most of our interview partners and half of the interviewed physicians knew LifeLines and PERT charts. LifeLines however, were understood much more easily when asking for

the possible meaning of an example.

When summarizing and evaluating the results of our user study, the following fundamental characteristics can be recognized – a simple and transparent structure, intuitive interaction (easy to learn and comprehend), a cleaned up interface, a high level of application safety (undo where possible), time saving (allowing quick and effective work), fast, and flexible.

The development of our visualization approach *CareVis* was primarily driven by the findings of this initial user study along with the results of our task and data analysis, and input from related work. The design and structure of CareVis' visualization and interaction methods is described in the following.

6 CareVis: Our Visualization Approach

The underlying data structure we want to communicate to medical domain experts is very complex. Since none of the examined visualization methods can be used to represent all needed data characteristics, we decided to use the approach of *multiple views*. Multiple views are a well known information visualization technique, whereby a number of representations that focus on different aspects of the data are provided for a common underlying data structure [21].

6.1 Views

Basically, we divided the underlying data structure along the lines of *logical structure* and *temporal aspects*. Hence, we provide a *Logical View* and a *Temporal View* along with a *QuickView Panel*. These distinct views are presented simultaneously and divide the screen in the following manner (see Fig. 2). The QuickView Panel is located on top of the screen displaying the most important patient parameters and plan variables at a selected position. Below that, the screen is divided vertically by the logical view on the left and the temporal view on the right side. The logical view presents treatment plans in terms of their logical structure (hierarchical decomposition, plan elements, execution order, conditions). The temporal view on the other side focuses on the temporal aspects of treatment plans and monitoring of measured patient data as well as plan variables (temporal aspects of plan elements, temporal uncertainties, hierarchical decomposition). Table 1 summarizes which data characteristics are visualized by the different views.

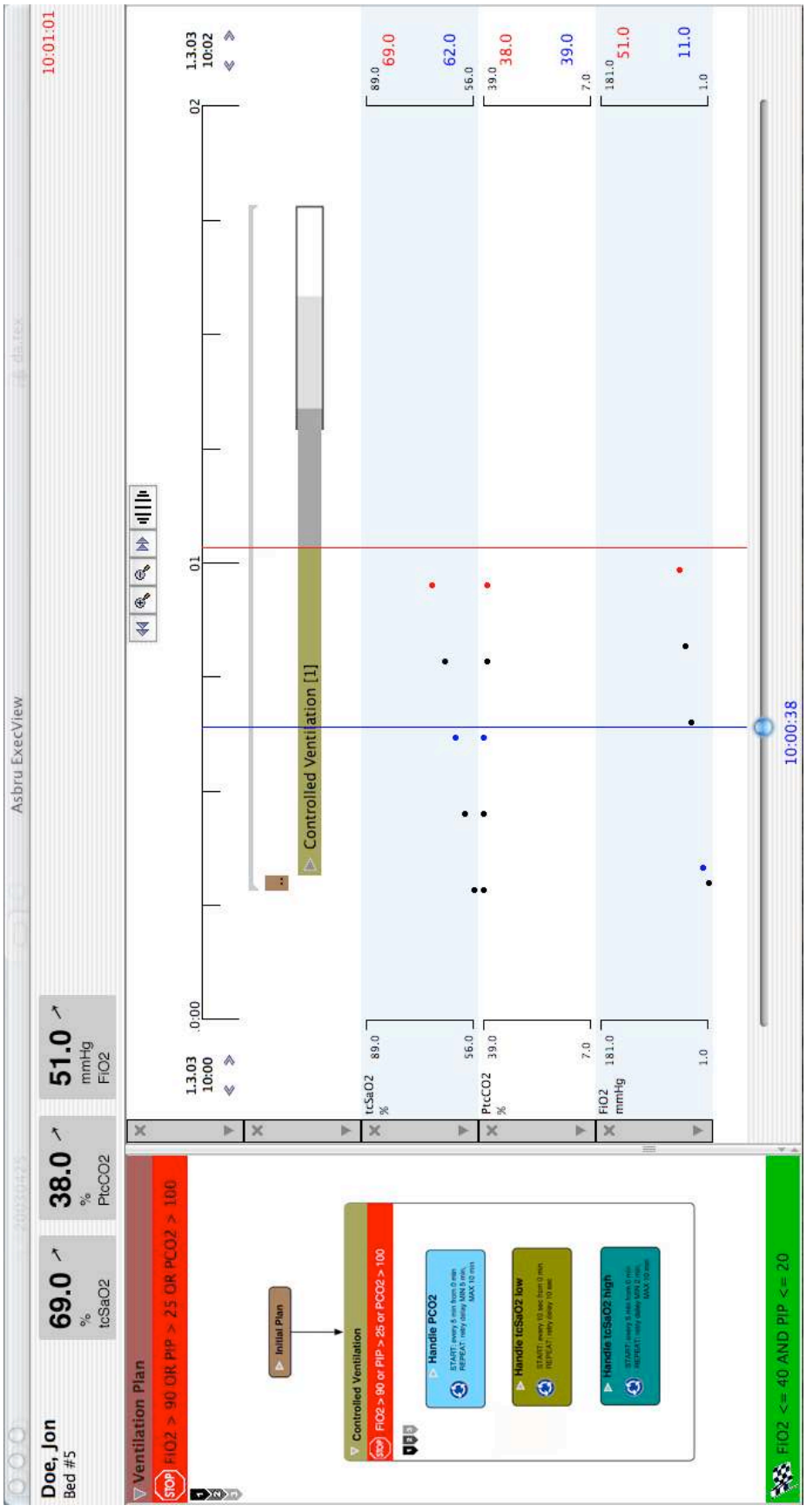


Fig. 2. Application window showing the execution of a plan (Mockup).

	Logical View	Temporal View	QuickView Panel
Asbru plans	•	•	
Time-oriented data		•	
Logical sequences	•		
Hierarchical decomposition	•	•	
Non-uniform element types	•	○	
Conditions	•		
Parameters and variables		•	•

entirely represented (•), partly/implicitely represented (○), or not represented (empty).

Table 1
Data characteristics in views.

6.1.1 Logical View

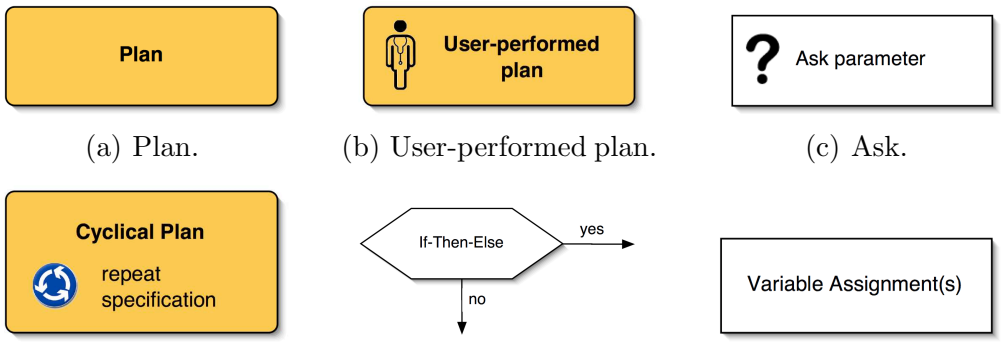
The logical view on the left-hand side of the screen provides a representation of the treatment plan specification data. The applied visualization technique *AsbruFlow* is based on the idea of flow-chart-like *clinical algorithm maps* [6] that are well known amongst physicians. This concept has been extended in order to be able to depict the characteristics of a treatment plan formulated in *Asbru*.

An *Asbru* plan has a plan-body containing *single-steps* that are executed in one of the execution orders *sequentially*, *parallel*, *any-ordered*, or *unordered*. A *single-step* is either a *variable assignment*, a *if-then-else* construct, an *ask* element, or a *plan activation*. Furthermore, an *Asbru* plan may contains three conditions: *filter precondition*, *abort condition*, and *complete condition*. These conditions are not just evaluated once at specific times during plan execution but are of state characteristic and thus checked all throughout the execution of the associated plan.

A set of six visual elements is used to depict the single steps within the body of an *Asbru* plan.

- *Plans* respectively *plan activations* are represented by a rounded rectangle filled with the plan color [4] (see Fig. 3(a)). In case of being a *cyclical plan*, an

[4] A distinct color is assigned to each plan, making it easier to distinguish plans from



(a) Plan. (b) User-performed plan. (c) Ask.

(d) Cyclical plan. (e) If-Then-Else. (f) Variable assignment.

Fig. 3. Plan step elements.

additional roundabout icon as well as the repeat specification in textual form are presented within the rectangle (see Fig. 3(d)). Furthermore, a physician icon appears within the element if the plan is *user performed* (see Fig. 3(b)).

- *Variable assignments* are represented by a rectangle containing the assignment textually (see Fig. 3(f)).
- *If-Then-Else* constructs are shown as hexagons having the condition displayed textually (see Fig. 3(e)).
- *Ask* steps of a plan are represented by a rectangle including a question mark ("?") symbol and the text "Ask" followed by the parameter to be entered into the system (see Fig. 3(c)).

For depicting plan conditions and the execution order of the plan steps, an enclosing frame was created (see Fig. 4). The topmost bar is filled with the plan color and contains the *title* of the plan. Below the plan title, the *abort condition* is shown. It is represented by a red bar having a stop sign icon at the left side. Right besides this icon, the abort condition is printed textually. The green bar at the bottom of the plan represents the *complete condition*. It has a checked finish flag icon at its left and contains the complete condition textually. The largest part of the representation is dedicated to the plan body of the depicted plan along with the *execution sequence indicator*. Its four possible symbols specify the execution order of the elements within the plan body – *sequentially*, *parallel*, *any-order*, or *unordered*.

The visual exploration of a treatment plan is supported by several interactive features. Plan elements that contain sub-elements are indicated by small gray triangles right in front of their labels. By clicking the triangle, the user navigates down the hierarchy, revealing the child elements of the chosen element. This navigational technique is well known from file system viewers as for example the *Finder* of the Macintosh™ system.

other elements and helping to recognize them in other parts of the representation.

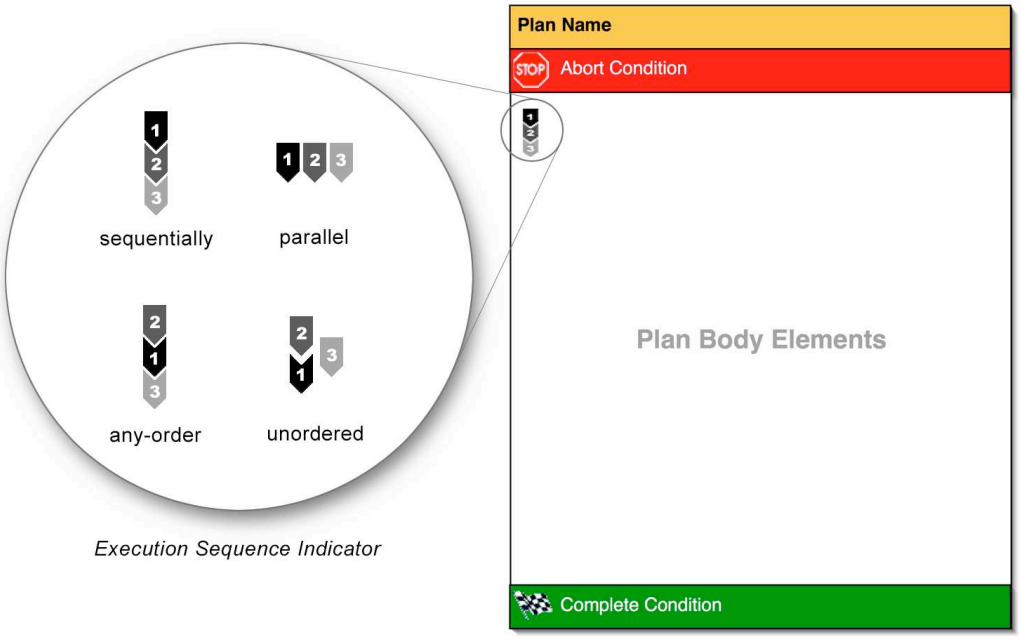
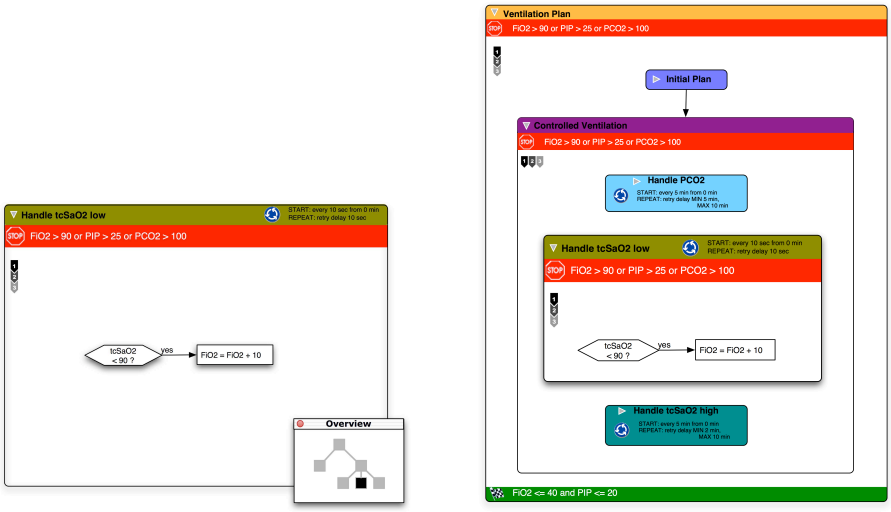


Fig. 4. Basic structure and execution sequence symbols.

In order to prevent getting lost within a plan by navigation, two *focus+context* techniques are applied. Firstly, there is the *overview+detail* technique that uses a small window containing a downscaled, simplified tree overview where the current position within a plan is highlighted. This small overview window can be toggled on or off (see Fig. 5, left). The second technique used is the *fisheye view* which distorts elements that are out of the current focus geometrically by shrinking and moving them (see Fig. 5, right). This method has been introduced by Schaffer et al. in their work on hierarchically clustered networks [22].

6.1.2 Temporal View

The *Temporal View* of our tool focuses on the time-oriented aspects of *Asbru* plans as well as displaying parameters and variables over time. This includes temporal attributes that are defined at design time and at run time. Design time attributes are either defined implicitly by execution order or explicitly via Asbru's time annotations. Run time attributes include plan execution times (plan start, plan end, etc.), plan variables over time, as well as measured patient data (parameters). The time interval covered by this view can span from a point in time in the past to a point in time in the future. Whereas only plans having time annotations (temporal planning attributes) can be displayed accurately in the future. Parameters and variables can only be displayed for past and present because they are only known and valid starting right at the point they appear.



(a) Overview+Detail. (b) Fisheye.

Fig. 5. Logical view showing parts of the Asbru plan for artificial ventilation of newborn infants (see Fig. 1) – Overview+Detail mode (left) vs. Fisheye mode (right).

The temporal view is divided into collapsable facets which can be added and removed dynamically (see Fig. 2). The most important element of this view is the time scale. It determines the portion of time being displayed. Below that, one facet is displayed containing the temporal aspects of the treatment plan elements followed by several facets containing different plan parameters and variables measured or computed over time. Collapsing facets leads to vertically shrunk and semantically zoomed representations which can be considered as *focus+context* technique (see Fig. 6(a) and Fig. 6(b)).

(a) Expanded view. (b) Collapsed view.

Fig. 6. Ventilation plan in expanded and collapsed view.

6.1.2.1 Time Scale The visible portion of time is determined by the *time scale*. It provides several interactive features for its manipulation. The viewed interval can be shifted forward or backward in time, zoomed in and out with automatic adjustment of the displayed ticks and labels, and begin and end point might also be manipulated independently. Moreover, a valuable feature offered by the time scale is the *fisheye display*. It magnifies a part of the scale interval (*focus*) while at the same time demagnifying the areas to the left and right of the focus (*context*) (see Fig. 7). This way, an area of interest is emphasized for detailed examination without hiding information before and

after that area, thus preserving the "full picture". The *focus+context* technique applied here is one having a non-continuous transformation function based on the *bifocal lens* [23].

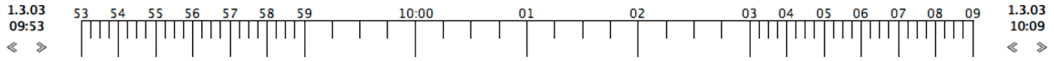


Fig. 7. Fisheye time scale.

6.1.2.2 Visualization of Plans The temporal representation of treatment plans is based on the idea of *LifeLines*. This concept has been extended for enabling the display of hierarchical decomposition as well as the complex time annotations used in *Asbru*. These new visual elements are called *LifeLines+* and *PlanningLines*, respectively. LifeLines+ allow the interactive representation of temporal intervals with hierarchical decomposition and simple element characteristics. On top of that, PlanningLines allow the depiction of temporal uncertainties via a glyph consisting of two encapsulated bars, representing minimum and maximum duration, that are bounded by two caps that represent the start and end intervals (see Fig. 9).

A *LifeLine+* has a defined beginning and a defined end represented visually by drawing a bar connecting those two points in time. The bar includes the title of the depicted incident plus a number of optional elements. *Exceed indicators* (small arrows) are displayed in case the LifeLine+ exceeds the shown interval as an indication that the currently visible line is only a part of the complete element. Furthermore, *property symbols* in form of small icons might be added on top of the line to indicate simple properties of the depicted incident (i.e. that the displayed plan is cyclical). To indicate hierarchical decomposition, small triangles are displayed in front of the line's caption in case the element contains child elements (analog to the *logical view*). By clicking this triangle, the element gets expanded. Here, the child elements are getting displayed as LifeLines+ and the expanded element itself is reduced to a gray, so-called *summary line*. By clicking onto this *summary line*, the reverse effect is triggered and the element gets collapsed into a LifeLine+.

PlanningLines are, as the name indicates, intended for depicting planning data afflicted with temporal uncertainties. Besides all aspects visualized by *LifeLines+*, *PlanningLines* offer additional support for the following rich set of time attributes:

- start interval (earliest starting shift + latest starting shift)
- end interval (earliest finishing shift + latest finishing shift)
- minimum duration
- maximum duration

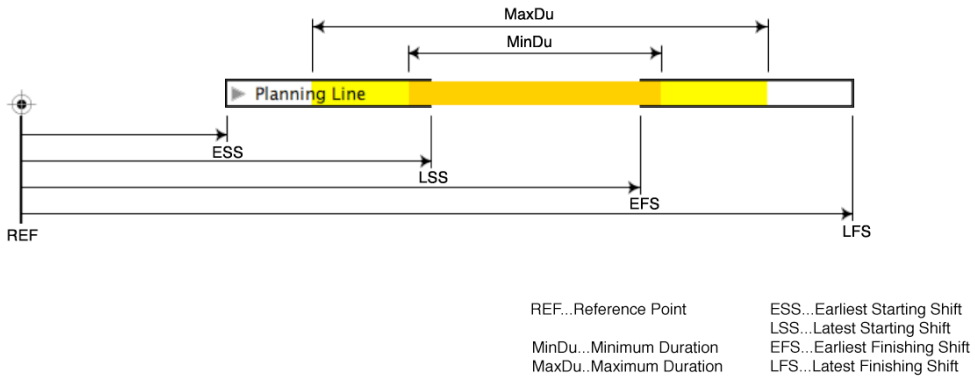


Fig. 8. PlanningLine.

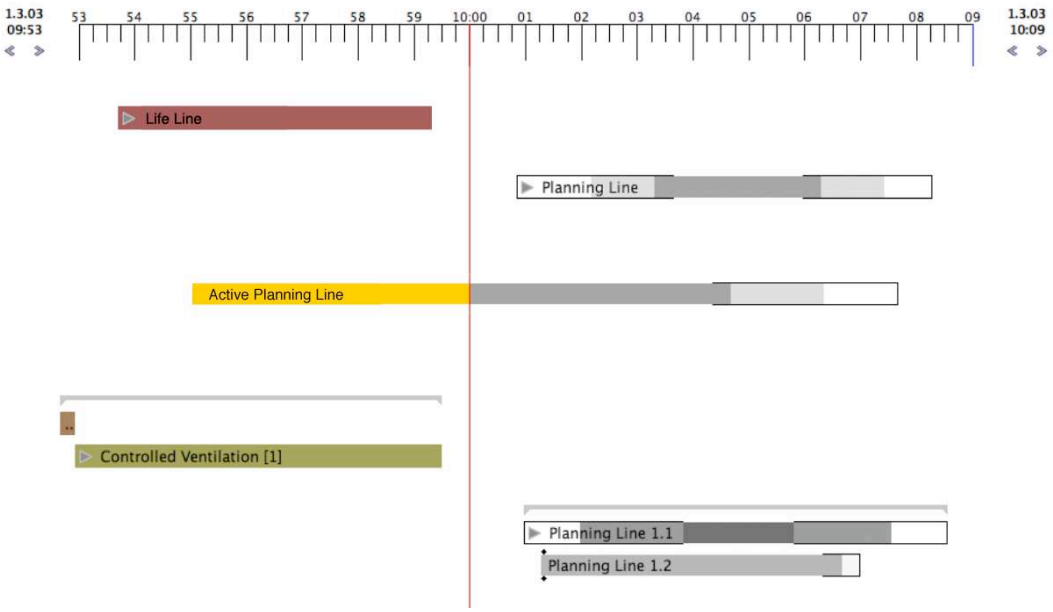


Fig. 9. Temporal view elements (LifeLines+, PlanningLines).

The glyph itself consists of three main parts: The *start cap* at the left, the *end cap* at the right, and the *duration bars* in between (see Fig. 8). The caps are drawn in black to emphasize their fixed position. The bars in contrary are colored whereas the color of the maximum duration bar has equal hue and saturation but higher brightness as the minimum duration bar. Encapsulated bars that can be shifted within the constraints of two mounted caps resemble the glyph's mental model.

6.1.2.3 Current Time Indicator and Time Cursor Two elements that have been proven to be very useful when interacting with the temporal view are the *Current Time Indicator* and the *Time Cursor*. The *Current Time Indicator* is a simple, vertical (in our case red) line, marking the current time on the one hand. Furthermore, the current time is displayed precisely in the upper right corner of the application window (see Fig. 2). The *Time Cursor* on the other hand marks an arbitrary point on the time scale. It is represented by a simple (in our case blue) vertical line and can be manipulated by mouse clicking and dragging (see Fig. 2). The precise value of the *Time Cursor* is displayed at the bottom of the application window right below the vertical time cursor line. This element might be used for example to inspect variable and parameter values at certain points in time, measure beginning and ending of plans, or compare various elements.

6.1.2.4 Visualization of Patient Data The facets below the temporal treatment plan representation are used for displaying measured patient data and plan variables. This work focuses on the integrative aspect and representing treatment plan information. Several novel approaches for visualizing time-oriented data that can be used for the graphical representation of patient data are described in [24].

6.1.3 QuickView Panel

A separate possibility to display currently valid variable and parameter values is the so-called *QuickView Panel* in the top part of the application window (see Fig. 2). The panel consists of rectangular areas that can be assigned to the available parameters and variables. A single item shows the current value along with its name, unit, and a trend indicator. Thus, the QuickView Panel allows to monitor the most important values by putting them at a prominent position, enlarged in size and without the need for displaying the complete history in an additional facet.

6.2 View coupling

Logical view and temporal view are tightly coupled in three different ways.

(1) A *common color palette* is used among the views for coloring plan elements.
(2) *Linking + brushing* through synchronous selection. If an element is selected in either the temporal or the logical view, the corresponding element(s) are selected in both views. This ensures a quick recognition and comparison of an element of interest in both views.

(3) *Navigation Propagation.* In contrast to the already presented methods, navigational procedures within a plan are not propagated to the coupled view, thus providing no automatic synchronization. Instead, view synchronization is user triggered via drag and drop. If the user wants to propagate the current position within a plan from one view to the other, she selects the desired element, moves it to the other view and drops it there. This user interaction initiates a navigation of the selected view to the desired position.

Figure 2 shows the *CareVis* application window during analysis of a ventilation plan. The "tcSaO2" facet indicates that the corresponding parameter is increasing. When referring to the PlanningLine display located above in the temporal view, we find that an instance of the "Controlled Ventilation" plan was performed while the parameter was increasing. To get more detailed information about this plan, we can drag the PlanningLine into the *AsbruFlow* panel (logical view) on the left-hand side, where the logical substeps of the plan are revealed.

7 Evaluation and Prototype

The designed methods have been discussed in a review step followed by the implementation of a Java prototype and its evaluation as described in the upcoming sections.

7.1 Design Review

When having completed the first "release" version of the conceptual design, we conducted a review session for getting early feedback regarding our design. This early evaluation process was very valuable and reduced the risk of investing time and effort in unfruitful initiatives.

The review was done qualitatively by two experts: one person is a visualization expert having experience in medical software development and the other one is a physician (medical expert) having visualization knowledge.

The result of the review was very positive, validating our concept, and showing that we were working in the right direction. Only some minor objections were raised about a small number of design issues. The suggestions were incorporated in an improved design (see [14] for detailed results).

7.2 Prototype Implementation

As a proof of concept and in order to generate a better impression of interaction issues, we implemented a Java prototype. For depicting the plan step elements in the flow-chart-like part of our representation, we used the graph drawing framework *JGraph* [25,26]. This is a flexible, small, and powerful package using the Model-View-Controller paradigm and is structured analogous to the standard *Swing* component *javax.swing.JTree*. All other graphical elements are embedded into the *Java Swing* standard component framework.

7.3 Prototype Evaluation

A scenario-based, qualitative prototype evaluation was carried out by conducting interviews with physicians working in intensive care units. Five of the eight physicians who already participated in the user study at the beginning of this work (see Section 5) took part in the evaluation. The interviews consisted of the four main parts: Introduction, Prototype Presentation, Prototype Testing, and Feedback/Questionnaire (refer to [14] for details).

The feedback regarding our design and prototype, given by the interviewed physicians, was very positive. All of them considered the overall structure clear, simple and not overloaded. The graphical representations and symbols have been judged to be intuitive and clear, keeping the learning effort relatively low. The interviewed doctors considered the two different views very helpful in working with and exploring treatment plans as well as patient data. Difficulties in relating the views to each other were not perceived.

A particular issue revealed by the prototype evaluation was that the navigation propagation interaction procedure proposed in the original design caused some confusion. Originally, a double-click initiated the navigation propagation which has been replaced by a more intuitive drag and drop interaction.

8 Supporting Users' Tasks

So how can Markus Zolte, Andrea Habacher, and Heinrich Kovanic benefit from our visualization methods in accomplishing their work tasks as described in Section 3.1?

8.1 Scenario 1

Markus Zolte wants to become acquainted with the hyperbilirubinemia protocol. Therefore, he loads the appropriate *Asbru* file and maximizes the logical view for examining the logical workflow of the plan. He uses the fisheye view for keeping an overview while exploring different paths of the plan using the small gray triangles for navigation through the hierarchy. When examining the intensive phototherapy part, he deactivates the fisheye view for displaying detail only and reads the abort condition in the read bar on top of the plan to identify cases in which the plan aborts. For getting positional information, he turns on the overview window. Finally, he opens a pull-down menu to see the full list of used parameters and variables.

8.2 Scenario 2

Andrea Habacher just completed treatment along the controlled ventilation plan and would now like to analyze the treatment history. She adjusts the zoom factor of the time scale for displaying the complete execution interval and explores the duration and position of the different phases. Furthermore, she uses the small gray triangles at the LifeLines+ to navigate to subplans. For investigating the "handle PCO2 plan", she selects the plan in the logical view and drops it into the temporal view. Subsequently, all instances of the plan are displayed and highlighted. Furthermore, she selects the PCO2 parameter at a pull-down menu for display in the temporal view. When encountering a significant discontinuity or the PCO2 value in one of the plan instances, she recalls the substeps of the plan by navigating down the hierarchy in the logical view to investigate which substeps of the treatment procedure might have caused this phenomenon. After that, she displays the "patient-state" parameter in the temporal view to examine how the PCO2 value influences it.

8.3 Scenario 3

Heinrich Kovanic is currently treating a patient along the hyperbilirubinemia protocol. He displays the "TSB" and "TSB-change" values in the temporal view as well as in the QuickView panel. In order to get basic patient information, he displays the patient record by double clicking the patient's name. After that, he displays all parameters and variables in the temporal view and encounters a rapid increase of the TSB value. He identifies the point in time of this episode by using the *time cursor*. He selects the plan that has been executed at that time in the temporal view and drops it into the logical view. The logical view navigates to the dropped plan and shows the details of the

applied parameter constraints defined by the plan conditions in the upper red and lower green bars. Finally, he double clicks the *current time indicator* at the upper right of the application window to navigate the temporal view back to the current position of plan execution.

9 Conclusion

Our goal was to develop visualization and interaction methods for supporting medical personnel in computerized protocol-based care. To achieve this goal, we had to consider several data aspects like the logic, structure, and temporal constraints of plans as given at design time, data of instantiated plans at execution time as well as patient data in form of parameters and variables. Several reasons led to the decision of introducing multiple simultaneous views for that matter. Applying a multiple views approach helped to master the complexity of the underlying data structure while using visualization methods well known to the domain experts. We have examined the usefulness of our approach performing a 3-step evaluation process including user study, design reviews, and prototype evaluation.

That visualizing the logic of clinical guidelines is useful to support understanding and exploration of protocols has already been proposed and proved years ago [7,6]. *Clinical algorithm maps* are most widely used in medical education and practice for that matter. This form of representation is clear, simple, and easily graspable – thus served as basis in our visualizations for the representation of a plan's logical structure. But it cannot be applied directly to represent *Asbru* plans because it does not provide a notion for representing hierarchical decomposition, flexible execution order, and state characteristics of conditions. Therefore, we extended this visualization by introducing new element types, an execution sequence indicator, and an enclosing frame containing the plan conditions.

Besides that, visualizing the temporal aspects of already executed plans, currently running plans, and plans to be executed in future in addition to the logic of treatment plans is vital for analysis and runtime support in medical treatment planning. Key issues of planning are temporal uncertainties inherently related to the temporal dimension. These uncertainties in the form of *Asbru's* powerful time annotations are visualized in a simple and meaningful way, fully integrated in the LifeLine based representation.

The use of software in contrast to paper allows us to support the process of exploring and understanding treatment plans at a higher level. It enables a meaningful navigation, providing annotations on demand for not overwhelming the viewer, and keeping orientation by using Focus+Context techniques,

thus increasing the flexibility in working with treatment plans. The introduced views focus on different aspects of the data while being tightly coupled to support physicians at their main work tasks.

An additional value, besides communicating plans to domain experts, became apparent during development. The visualization of plans helps to spot problems, bugs, and ambiguities in the formal plan representation which are hard to see and detect otherwise. Furthermore, the visualization serves as an important basis for the communication between medical domain experts and computer scientists.

Moreover, we applied a user-centric approach when developing our visual representation – we involved the end-users from the very beginning by carrying out a user study and evaluated our design as well as our prototype. This increases the quality of design, the user acceptance, and serves as an indicator of the maturity of development. We used a well known graphical representation as basis and introduced a cleaned up interface that has a simple and transparent structure with only a handful of different visual elements which are easy to learn and comprehend. The interaction is carried out intuitively by applying well known techniques from standard software supported by different focus+context techniques for keeping an overview. The most important user requirement of being time-saving is achieved by combining intuitive navigation and rich information presentation in a structured way. This is in contrast to working with paper-based treatment protocols and patient records that are often a mix of text, tables, and graphics, scattered over various pages, making it hard to keep an overview and conceive the logic of a treatment plan.

Acknowledgements

This project is supported by "Fonds zur Förderung der wissenschaftlichen Forschung - FWF" (Austrian Science Fund), grant P15467-INF.

References

[1] M. Field, K. Lohr, Guidelines for Clinical Practice: From Development to Use, Institute of Medicine, Washington, D.C. National Academy Press, 1992.

[2] M. Peleg, S. Tu, J. Bury, P. Ciccarese, J. Fox, R. Greenes, R. Hall, P. Johnson, N. Jones, A. Kumar, S. Miksch, S. Quaglini, A. Seyfang, E. Shortliffe, Stefanelli, Comparing Computer-Interpretable Guideline Models: A Case-Study Approach, The Journal of the American Medical Informatics Association (JAMIA) 10 (1) (2003) 52–68.

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

[4] S. Miksch, Plan Management in the Medical Domain, AI Communications 12 (4) (1999) 209–235.

[5] A. Cooper, The Inmates Are Running The Asylum: Why High Tech Products Drive Us Crazy and How To Restore The Sanity, SAMS Publishing, 1999.

[6] D. C. Hadorn, Use of Algorithms in Clinical Practice Guideline Development: Methodology Perspectives, AHCPR Pub. 0009 (95) (1995) 93–104.

[7] Society for Medical Decision Making, Proposal for Clinical Algorithm Standards, Medical Decision Making 12 (02) (1992) 149–154.

[8] R. Kosara, S. Miksch, Metaphors of Movement — A User Interface for Manipualting Time-Oriented, Skeletal Plans, Artificial Intelligence in Medicine 22 (2) (2001) 111–132.

[9] R. Kosara, S. Miksch, Visualizing Complex Notions of Time, in: J. Roberts (Ed.), Proceedings of the Conference on Medical Informatics (MedInfo 2001), 2001, pp. 211–215.

[10] E. Tufte, S. M. Powsner, Graphical Summary of Patient Status, The Lancet 344 (8919) (1994) 386–389.

[11] C. A. Brandt, S. J. Frawley, S. M. Powsner, R. N. Shiffman, P. L. Miller, Visualizing the Logic of a Clinical Guideline: A Case Study in Childhood Immunization, Methods of Information in Medicine 36 (1997) 179–83.

[12] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, B. Shneiderman, LifeLines: Using Visualization to Enhance Navigation and Analysis of Patient Records, in: Proceedings of the 1998 American Medical Informatic Association Annual Fall Symposium, 1998, pp. 76–80.

[13] www.openclinical.org, Open Clinical - Knowledge Management for Medical Care, http://www.openclinical.org (2003).

[14] W. Aigner, Interactive Visualization of Time-Oriented Treatment Plans and Patient Data, Master's thesis, Vienna University of Technology, Institute of Software Technology and Interactive Systems, Vienna, Austria (May 2003).

[15] B. Johnson, B. Shneiderman, Treemaps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures, in: Proceedings of the IEEE Information Visualization '91, IEEE, 1991, pp. 275–282.

[16] C. Plaisant, B. Milash, A. Rose, S. Widoff, B. Shneiderman, LifeLines: Visualizing Personal Histories, in: Proceedings CHI'96 ACM Conference on Human Factors in Computing Systems, ACM Press, New York, 1996, pp. 221–227.

[17] C. Combi, L. Portoni, F. Pinciroli, Visualizing temporal clinical data on the www, in: W. Horn, Y. Shahar, G. Lindberg, S. Andreassen, J. Wyatt (Eds.), Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making (AIMDM'99), Springer, 1999, pp. 301–311.

[18] L. Chittaro, C. Combi, Visual Definition of Temporal Clinical Abstractions: A User Interface Based on Novel Metaphors, in: Proceedings of AIME 01: 8th Conference on Artificial Intelligence in Medicine Europe, Lecture Notes in Computer Science, Vol. 2101, 2001, pp. 227–230.

[19] P. Messner, Time Shapes - A Visualization for Temporal Uncertainty in Planning, Master's thesis, Vienna University of Technology, Institute of Software Technology and Interactive Systems, Vienna, Austria (April 2000).

[20] R. Kosara, P. Messner, S. Miksch, Time and Tide Wait for No Diagram, Tech. Rep. Asgaard-TR-2001-2, Institute of Software Technology and Interactive Systems, Vienna University of Technology, Austria (2001).

[21] J. C. Roberts, On Encouraging Multiple Views for Visualization, in: E. Banissi, F. Khosrowshahi, M. Sarfraz (Eds.), IV'98 – Proceedings International Conference on Information Visualization, IEEE Computer Society, 1998, pp. 8–14.

[22] D. Schaffer, Z. Zuo, S. Greenberg, L. Bartram, J. Dill, S. Dubs, M. Roseman, Navigating Hierarchically Clustered Networks through Fisheye and Full-Zoom Methods, ACM Transactions on Computer-Human Interaction 3 (2) (1996) 162–188.

[23] Y. K. Leung, M. D. Apperley, A review and taxonomy of distortion-oriented presentation techniques, ACM Transactions on Computer-Human Interaction 1 (2) (1994) 126–160.

[24] R. Bade, S. Schlechtweg, S. Miksch, Connecting Time-Oriented Data and Information to a Coherent Interactive Visualization, in: Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI 2004, ACM Press, 2004, pp. 105–112.

[25] G. Alder, Design and Implementation of the JGraph Swing Component, Tech. Rep. 1.0.6 (February 2002).

[26] G. Alder, The Home Page of JGraph, http://jgraph.sourceforge.net (2002).