Temporal Dimension of Medical Guidelines: The Semantics of Asbru Time Annotations

Jonathan Schmitt,[1] Wolfgang Reif,[1] Andreas Seyfang,[2] Silvia Miksch[2,3]

Abstract. The temporal dimension of observations, actions, and tasks described in clinical guidelines and protocols (CGP) is important and at the same time sometimes vague or complex. Correct understanding of a modelling language of CGPs is basis for a high quality formal model.

In this paper, we describe how the temporal dimension is specified using Asbru time annotation as well as the semantics and some important properties of these time annotations.

1 Introduction

Clinical guidelines and protocols (CGPs) are an important means to ensure the quality of medical care, to aid the medical staff and to control cost. In order to use computer support for verification and execution, they need to be translated to a formal representation such as Asbru. Once formalised, properties to ensure the soundness of CGPs can be verified, which is an important method to ensure integrity and quality.

The temporal aspect of both observations (patient state) and actions (treatment steps) is important in practical applications of CGPs. An example for the temporal dimension of a patient state observation is the measurement of bilirubin in the treatment of Jaundice in otherwise healthy newborns [1]. Here, the treatment depends on the level of bilirubin in the blood and on the age of the infant. This age is abstracted to the three possibilities first day, second day and later. When translating this guideline to a formal representation such as Asbru, it is necessary to clarify this temporal information: Is the first day the calendar day, on which the baby was born, or is it the first 24 hours after birth? In practice the care personnel may use their own rules to map the term in the guideline to their actual actions, potentially deviating from the intended treatment. Therefore, these details must be clarified and expressed in an unambiguous way when formalising the guideline.

While adding such information causes effort on the part of the modeller, the resulting model provides valuable feedback for the original guideline. In particular, the now explicit time constraints can be discussed by domain experts and verified using formal methods. Also, a precise model is a precondition for executing the guideline in a decision support system, which not only is an important means to implement the guideline in practice, but also helps to "debug" the guideline by testing various scenarios, feeding it different sets of patient data.

Among a range of formal representations of CGPs, Asbru was found to have the strongest features to express temporal aspects [2]. It is a time-oriented plan representation language that represents clinical guidelines as skeletal plans. In Asbru, the concept of skeletal plans has been enriched with temporal aspects.

Asbru was originally developed by the Asgaard project [3]. Later Asbru evolved into an XML-based language with extensions regarding temporal data abstraction and more specific modelling of data [4]. The semantics of a subset of Asbru has been defined using Structured Operational Semantics [5].

All conditions for the transition from one plan state to another are expressed in terms of temporal patterns. A temporal pattern consists of one or more parameter propositions or plan-state descriptions. Each parameter proposition contains a value description, a context, and a time annotation.

A time annotation describes sets of time intervals in a flexible way. It specifies a time range for the starting point, one for the finishing point and one range restricting the duration of the intervals. Start and end ranges are defined relative to a reference point. Using different reference points in different time annotations allows for different time lines in the CGP. Time ranges may be partly undefined – see Table 1 for details.

In practical knowledge acquisition, the variety of values used to describe a single interval often confuses users. The main reason is that in daily life we deal with abstracted descriptions which refer to the typical case and which imply some tolerance regarding actual start, end and duration.

To overcome this problem, it has proven useful to query each of the 6 constraints on its own. Doing so, taking the negative approach yields more precise results. I.e., if you ask "should an interval be included that starts after X?" is perceived as a weaker statement than "should an interval be excluded that starts before X?". In other words, each of the 6 constraints defines an exclusion criterion for intervals to match, but only all together they describe the inclusion criterion. Therefore, looking at isolated values, one must think about exclusion instead of inclusion.

The same question about the exclusion gives the answer to whether this value is used at all. If the question "When is the duration of this interval too short?" cannot be answered, then the minimum duration obviously is not defined here. In contrast, the answer to a more general question like "How long should the interval be?" will not directly map to a time annotation.

[1] University of Augsburg, D-86135 Augsburg, Germany, {schmitt, reif}@informatik.uni-augsburg.de
[2] Vienna University of Technology, A-1040 Vienna, Austria, {seyfang, silvia}@asgaard.tuwien.ac.at
[3] Danube-University Krems, A-3500 Krems, Austria

Name	Abbr.	Default	Description
Reference point	RP		The time point, to which the following four values (called *shifts*) are offsets.
Earliest starting shift	ESS	$-\infty$	The earliest point in time, at which a fitting interval may start.
Latest starting shift	LSS	∞	The latest point in time, at which a fitting interval may start.
Earliest finishing shift	EFS	$-\infty$	The earliest point in time, at which a fitting interval may end.
Latest finishing shift	LFS	∞	The latest point in time, at which a fitting interval may end.
Minimum duration	MinDu	0	The shortest time span of an interval to be a fitting interval.
Maximum duration	MaxDu	∞	The longest time span of an interval to be a fitting interval.

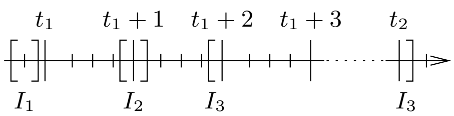
Table 1. Elements of a time annotation.

2 Timing in Asbru

Timing in Asbru is based on the strong synchronous hypothesis [6]. This basically means that system transitions are assumed to be instantaneous, while only environmental transitions use up time. This is implemented by a micro/macro step timing scheme, with system transitions happening during the micro steps while environment transitions happen during the macro steps. Macro steps are only allowed to occur, if the system state is stable, which is, if all active system transitions cannot change the state of the system, i.e. the system stutters.

Input variables, e.g. patient data, do not change during micro steps. Internal variables, e.g. the plan states, may change during a micro step. Time proceeds only, if a macro step happens. The duration between two points in time, t_1 and t_2 is calculated as t_2 - t_1 and is zero if $t_2 = t_1$.

Timing intervals can be specified ranging over a number of micro and macro steps. A timing interval [t_1, t_2] with $t_1 < t_2$ contains the macro step at t_1 and all micro and macro steps from t_1 + 1 to t_2 - 1, if there are any. A timing interval [t, t] contains some of the micro steps which happened before the macro step t. For examples on the duration see Figure 1.



I_1	$[t_1, t_1]$	duration 0
I_2	$[t_1+1, t_1+2]$	duration 1
I_3	$[t_1+2, t_2+1]$	duration $t_2-(t_1+1)$

Figure 1. Intervals in Asbru

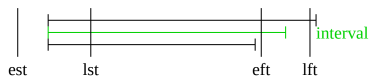


Figure 2. Time annotation example

3 Formal Definition of Time Annotations

3.1 Introduction

Figure 2 shows the visualisation of an Asbru time annotation. Asbru time annotations are representations of sets of intervals. One of the intervals, which is element of the set of intervals being defined by the time annotation is depicted by the central horizontal line, marked "interval". The lower horizontal line represents the minimum length of the intervals defined by minDu, the upper horizontal line the maximum length of the intervals specified by maxDu in the time annotation.

As can be seen, the interval is starting within the starting interval (i.e. between the lines marked with EST and LST), ends within the finishing interval (i.e. between the lines marked with EFT and LFT) and its length is between the length of minDu and maxDu. EST is equal to ESS + RP and so on; see below for details.

In the next section we will start defining time annotation as well as some predicates necessary to reason about them.

3.2 Formal Definition

Definition (infinite numbers) $\widehat{\mathbb{Z}}$ is the set of integers, united with positive and negative infinity, $\widehat{\mathbb{Z}} = \mathbb{Z} \cup \{\infty, -\infty\}$

Definition (interval) An interval is a pair, $[a, b]$, where $a, b \in \widehat{\mathbb{Z}}$, and $a \le b$. A variable c with $c \in \widehat{\mathbb{Z}}$ is element of an interval, $c \in [a, b]$, iff $a \le c$ and $c \le b$.

Definition (time annotation) A time annotation is a four tuple, syntactically written

$$[\text{ESS, LSS}][\text{EFS, LFS}][\text{minDu, maxDu}]\ \text{RP}$$

where ESS, EFS $\in \mathbb{Z} \cup \{-\infty\}$, LSS, LFS, maxDu $\in \mathbb{Z} \cup \{\infty\}$, minDu, RP $\in \mathbb{Z}$ and [ESS, LSS], [EFS, LFS] and [minDu, maxDu] are pairs. The following functions are defined on time annotations:

$TA\ .ess$	= ESS	$TA\ .est$	= ESS + RP (= EST)
$TA\ .lss$	= LSS	$TA\ .lst$	= LSS + RP (= LST)
$TA\ .efs$	= EFS	$TA\ .eft$	= EFS + RP (= EFT)
$TA\ .lfs$	= LFS	$TA\ .lft$	= LFS + RP (= LFT)
$TA\ .maxDu$	= maxDu	$TA\ .sti$	= [$TA\ .est$, $TA\ .lst$]
$TA\ .minDu$	= minDu	$TA\ .fti$	= [$TA\ .eft$, $TA\ .lft$]
$TA\ .refPoint$	= RP	$TA\ .dti$	= [minDu, maxDu]

Definition (matching) A time annotation defines a set of intervals, and an interval [a, b] is matched by a time annotation TA, written as [a, b] $\in TA$, iff a $\in TA\ .sti$, b $\in TA\ .fti$ and b - a $\in TA\ .dti$.

Definition (legal) A time annotation TA is legal, iff the following holds:

$$TA\ .minDu \leq TA\ .lfs - TA\ .ess$$
$$TA\ .efs - TA\ .lss \leq TA\ .maxDu$$

Theorem (legal-member) A time annotation TA is legal iff there is an interval I with $I \in TA$.

Comment (legal-member) The predicate legal offers a simple possibility to determine, whether there is at least one interval, which is matched by the time annotation. Time annotations which are not legal – and therefore have no matching interval – usually indicate a modelling errors, with the exclusion criteria being to strong.

Definition (normal) A time annotation TA is normal, iff the following holds:

$$TA\ .efs - TA\ .lss \leq TA\ .minDu$$
$$TA\ .efs - TA\ .maxDu \leq TA\ .ess$$
$$TA\ .maxDu \leq TA\ .lfs - TA\ .ess$$
$$TA\ .lfs \leq TA\ .maxDu + TA\ .lss$$
$$TA\ .minDu + TA\ .ess \leq TA\ .efs$$
$$TA\ .minDu + TA\ .lss \leq TA\ .lfs$$
$$0 \leq TA\ .minDu$$

Theorem (normal) A time annotation TA is normal, iff all of the following holds:

For all a $\in TA\ .sti$, there exists b such that [a, b] $\in TA$.
For all b $\in TA\ .fti$, there exists a such that [a, b] $\in TA$
For all c $\in TA\ .dti$, there exist a, b such that
  b - a = c and [a, b] $\in TA$

Comment (normal) The theorem normal establishes, that a normal time annotation is minimal. In Fig. 3, the maximum duration maxDu is larger than the difference between the LFT and the EST. An interval matching the time annotation is required to start within the starting shift (i.e. no sooner than EST) and to finish in the finishing shift (i.e. no later than LFT). This way, a duration larger than maxDu' is not possible to achieve for a matching interval. In this sense, the example is not minimal and therefore should not be normal, which it is not, normal requires the maxDu to be smaller than LFS - EFS. Typically such situations occur, if the modeller is asked all six values of time annotations independently, as is suggested in Section 1. Although a maximum duration is not specified by the modeller, it may be given implicitly due to constraints regarding the earliest starting point and the latest finishing point. In such cases it is wanted to write down a time annotation just as the modeller specified, because otherwise the constraints would be harder to understand for humans. At the same time having the time annotation in a minimal way, as is defined by the predicate normal, allows for a more efficient execution. The usual mode of operation is therefore to allow the specification of legal but non-normal time annotations within Asbru and decision support systems for an improved understanding, but normalise them for the use within the decision support or verification system to be able to use more efficient implementations relying on the minimality.

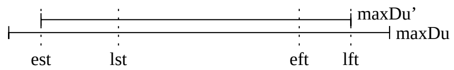


Figure 3. Example of a non-normal time annotation

Theorem (normal - legal (1)) Any legal time annotation TA can be rewritten into a time annotation TA', where TA' is normal and for all intervals I, it holds that $I \in TA$ iff $I \in TA'$.

As a proof for this theorem, we provide a translation scheme and verify, that for every time annotation TA', generated by this translation scheme from a legal time annotation TA, it holds that TA' is normal and that for all intervals I, I $\in$ TA' iff I $\in$ TA.

$$TA'\ .ess = \max(TA\ .ess\ ,\ TA\ .efs\ -\ TA\ .maxDu\ )$$
$$TA'\ .lss = \min(TA\ .lss\ ,\ TA\ .lfs\ -\ TA\ .minDu\ )$$
$$TA'\ .efs = \max(TA\ .efs\ ,\ TA\ .ess\ +\ TA\ .minDu\ )$$
$$TA'\ .lfs = \min(TA\ .lfs\ ,\ TA\ .lss\ +\ TA\ .maxDu\ )$$
$$TA'\ .minDu = \max(TA\ .minDu\ ,\ TA\ .efs\ -\ TA\ .lss\ )$$
$$TA'\ .maxDu = \min(TA\ .maxDu\ ,\ TA\ .lfs\ -\ TA\ .ess\ )$$
$$TA'\ .refPoint = TA\ .refPoint$$

Comment (normal - legal (1)) As stated above, this translation pattern can be used to normalise legal time annotations. Using this pattern in an appropriate way allows to specify all time annotations in a legal but non-normal way, which may be better understandable by an Asbru modeller. At the same time, the efficiency of the implementation of the verifier or interpreter can be greatly improved, if it is known, that only normal time annotations have to be dealt with in a case study.

Theorem (normal - legal (2)) Every normal time annotation is also legal.

Verification of the presented theorems All theorems and the reduction scheme in this section have been verified using the interactive theorem prover KIV [7]. During verification, errors in the original formulation of the definition normal were found regarding special cases involving infinite numbers. Consequently, the definitions were corrected. All theorems can be verified with the definition of normal and legal as presented in this paper.

Although the time annotations are used in temporal logic verification work, we decided to specify the semantics using first order logic. It seemed to us more fitting to do so, as a specification using temporal logic would add complexity while providing no additional benefit, as reasoning about the length of intervals is very well covered by first order logic.

4 Conditions

Conditions in Asbru are used as guards to internal transitions, evaluated over medical data of the patient. Conditions come in two flavours. Basic conditions map a state to a truth value. Complex conditions combine an Asbru condition, called a conditional, with a time annotation. Complex conditions determine, whether an interval can be found which matches the time annotation and during which the conditional is evaluated to true. For the evaluation of complex conditions it is usually necessary to use history variables to keep track of the past states the system has reached.

In the context of complex conditions, it can be said, that the conditional is specifying the desired behaviour, while the time annotation specifies the timing aspects. Evaluation of the truth value of complex conditions is threefold. A complex condition is true, iff an interval as described above can be found. A complex condition is false, iff no such interval can be found and it can be derived, that expanding the histories into the future will not result in a situation where the condition is true. If the condition is currently not true but could become true in the future, the status of the condition is unknown.

4.1 Example

Assume an infection with measles. After contact with a measles infected person, inflammation of the mucous membranes is a warning sign for an infection. In cases of such inflammations, the patient should be quarantined to prevent a spreading of the disease. In Asbru this could be formulated as a complex condition with the conditional being "inflammation of the mucous membranes" and the time annotation

$[8\text{ days}, 12\text{ days}][-\infty, \infty][0, \infty]$ **contact to infected person**

It is obvious that prior to the 8^{th} day after the contact to an infected person, it cannot be decided, whether or not a measles related inflammation will occur. Therefore, the condition will evaluate to unknown during this period.

During the period of eight to twelve days after the contact, any occurring inflammation of the mucous membranes will make the condition true immediately, resulting in a plan to be started to quarantine the patient and to check for a measles infection.

Assume, an inflammation starts only one day after the contact to a measles infected person. In that case, it is obvious, that the reason is not this specific contact to the infected person. Therefore this infection, should it last into the 8/12 day interval, does not trigger the quarantine. Instead, the requirement is, that there is an infection within 8 to 12 days and this infection started no sooner than 8 days after contact.

Twelve days after the contact to an infected person without an inflammation of the mucous membranes, it can be decided, that an infection has not occurred and the condition will evaluate to false.

A more cautious approach would be to state, that it is not necessary for the inflammation to start later than 8 days after contact, but only to happen then. The time annotation of the example would then be changed to

$[-\infty, 12\text{ days}][8\text{ days}, \infty][0, \infty]$ **contact to infected person**

The decision about the correctness of these time annotations can only be made in coordination with a domain expert.

4.2 Flank behaviour

In this section we will provide some examples and describe the effects that may result from a wrong formulation of time annotations. As discussed in the previous section, certain time annotations require the behaviour not only to occur, but to start or stop within a certain interval. If the ESS is not set to $-\infty$ or the maxDu is set to ∞, a start in behaviour has to be observed in the starting interval. If the LFS or the maxDu are not set to ∞ an end of the behaviour in the finishing interval has to be observed.

As an example, we will use a complex Asbru condition, where some conditional is paired with the time annotation [4h, ∞][-∞, 6h][0, ∞] **self**. Self refers to the start of the plan containing this time annotation. From the description in the previous paragraph, it can be seen, that a raising flank in the beginning is required, because the ESS is set to 4h. A raising flank is a observed, if the specified behaviour of a condition could not be observed at one time point, but could be observed in the very next time point. A trailing flank in the ending is required, because the LFS is set to 6h. A trailing flank is defined inversely to a raising flank.

Intervals satisfying this time annotation are all those starting not earlier than 4 hours after the **self** time and ending not later than 6 hours after the **self** time. Some example traces for this time annotation are depicted in Figure 4. As can be seen in the first two traces, an interval can be found such that the interval is member of the time annotation. Also, the flank requirements are satisfied with these intervals. The third example displays a trace, where the condition is not satisfied, because no trailing flank in the finishing interval can be observed.

From the point of view of a decision support system, it is also important to determine the point in time, where a time annotated condition becomes true. This has been visualised in Figure 5, where two points are marked up, the first, where

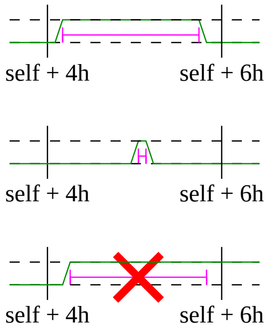


Figure 4. Example traces

the conditional, the truth mapping disregarding the time annotation becomes true. The second point is the time, where the condition, that is, the conditional including the time annotation, becomes true.

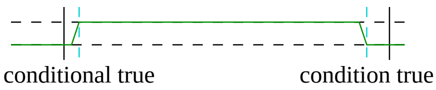


Figure 5. Example traces

Obviously, the existence of an interval with an accompanying trailing flank in the end can only be determined after this flank has been observed. If flanks are not required, a time annotation has to be found, without ESS, LFS and maxDu being set. Such a time annotation for the above example is [-∞, 6h][4h, ∞][0, ∞] **self**, which requires the conditional to be true sometime during the interval [**self** + 4h, **self** + 6h].

4.3 History variables

To evaluate complex conditions, it is usually necessary to keep track of the past states a system had reached. For this, history variables are employed.

Variables may be assigned different values for each micro step. This has to be reflected in the history variables, such that all assumed values have to be stored. Additionally, the order of occurrence has to be stored. All the assumed values of a variable are stored in a list, ordered by actuality. The last entry is the most current one before a macro step.

4.4 Formal semantics of conditions

Simple Asbru conditions are first order logic formulas φ, mapping a state σ to a truth value, $\varphi(\sigma)$. Complex Asbru conditions are pairs of Asbru conditions, called conditional, and time annotations. Their truth evaluation is more complex and requires some additional definitions.

Definition (state) A state σ is a mapping of variables to values. Each state is time-stamped, where the time stamp of a state σ can be retrieved using the selector $\mathrm{t}(\sigma)$. The selector p maps a state σ to a natural number.

Definition (history) A history is a function, mapping a list of states to a time t, $\mathrm{t} \in \mathbb{Z}$. The selector h[t] selects the list of states associated with time t in history h. A state σ is in a history, iff there exists t such that $\sigma \in \mathrm{h}[\mathrm{t}]$.

Within a history it must hold for all t, that h[t] is nonempty and if $\mathrm{h}[\mathrm{t}] = [\sigma_1, \sigma_2, \ldots, \sigma_n]$ then for all σ_i, $\mathrm{t}(\sigma_i) = \mathrm{t}$ and $\mathrm{p}(\sigma_i) = \mathrm{i}$.

Definition (successor, predecessor) A state σ_2 is the successor of a state σ_1, written as $\sigma_1 < \sigma_2$, iff $\mathrm{t}(\sigma_1) < \mathrm{t}(\sigma_2)$ or $\mathrm{t}(\sigma_2) = \mathrm{t}(\sigma_1)$ and $\mathrm{p}(\sigma_1) < \mathrm{p}(\sigma_2)$. $\sigma_1 \leq \sigma_2$, iff $\sigma_1 < \sigma_2$ or $\sigma_2 = \sigma_1$.

A state σ_2 is the direct successor of a state σ_1, written as $\mathrm{succ}(\sigma_1) = \sigma_2$, iff $\sigma_1 < \sigma_2$ and for all σ_i, $\sigma_i < \sigma_2$ it holds that $\sigma_i \leq \sigma_1$. A state $\sigma_1$is the direct predecessor of a state σ_2, written as $\mathrm{pred}(\sigma_2) = \sigma_1$, iff $\mathrm{succ}(\sigma_1) = \sigma_2$.

Definition (eval) The function eval maps an Asbru condition cond, a history h and a state σ to {`true`, `false`, `unknown`}. For a simple Asbru condition φ, $\mathrm{eval}(\varphi, \mathrm{h}, \sigma) =$ `true`, iff $\varphi(\sigma)$; $\mathrm{eval}(\mathrm{cond}, \mathrm{h}, \sigma) =$ `false`, iff not $\varphi(\sigma)$.

Definition (satisfied) Satisfied maps an Asbru condition, a history and a state to a Boolean value. satisfied((condition $\times$ *TA*), h, σ), iff there exists states σ_1, σ_2, such that $\sigma_1, \sigma_2 \in \mathrm{h}$, $[\mathrm{t}(\sigma_1), \mathrm{t}(\sigma_2)] \in TA$, $\sigma_1 < \sigma_2 \leq \mathrm{succ}(\sigma)$ and for all σ_n, $\sigma_1 \leq \sigma_n$ and $\sigma_n < \sigma_2$ eval(condition, h, σ_n) = `true`, if $\sigma_n \in \mathrm{h}$.

If *TA .ess* is not set to $-\infty$ or *TA .maxDu* is not set to ∞, furthermore eval(condition, h, pred(σ_1)) = `false`and if *TA .lfs* or *TA .maxDu* are not set to ∞, eval(condition, h, σ_2) must be `false` and $\sigma_2 \leq \sigma$.

Definition (satisfiable) Satisfiable is a predicate mapping an Asbru condition, a history and a state to a Boolean value. satisfiable((condition $\times$ *TA*), h, σ), iff there exist states σ_1, σ_2, such that $\sigma_1 \in \mathrm{h}$ or $\sigma < \sigma_1$, $\sigma_2 \in \mathrm{h}$ or $\sigma < \sigma_2$, $[\mathrm{t}(\sigma_1), \mathrm{t}(\sigma_2)] \in TA$, $\sigma_1 < \sigma_2$ and for all σ_n, $\sigma_1 \leq \sigma_n$ and $\sigma_n < \sigma_2$ and $\sigma_n \leq \sigma$, eval(condition, h, σ_n) = `true`.

If *TA .ess* is not set to $-\infty$ or *TA .maxDu* is not set to ∞, it must hold that eval(condition, h, pred(σ_1)) = `false` or pred(σ_1) $< \sigma$. If the *TA .lfs* or *TA .maxDu* are not set to ∞, eval(condition, h, σ_2) must be `false` or $\sigma < \sigma_2$.

Definition (complex-eval) Given a history h, a state σ and an complex Asbru condition cond, eval(cond, h, σ) = `true`, iff satisfied(cond, h, σ), eval(cond, h, σ) = `false`, iff not satisfiable(cond, h, σ), eval(cond, h, σ) = `unknown`, iff satisfiable(cond, h, σ) but not satisfied(cond, h, σ).

5 Discussion and Conclusion

The temporal dimension of CGPs is receiving increasing attention.

GLIF3, (New)GUIDE, PRODIGY, PROforma [2] and GLARE [8] handle time like other values in conditions. This is a suitable solution for simple cases in low-frequency domains. However, combining several constraints on a single interval leads to complex conditions. Also efficient monitoring as described in [9] is not possible.

The time annotation used in Asbru is based on work of Dechter et al. [10] and Rit [11]. Being a primary element of the language, it is handled completely different than value descriptions which refer to the value of measurements. This yields two advantages: The interpreter can schedule the monitoring efficiently, avoiding "busy wait"; and clarifications concerning the interdependencies of different temporal constraints become part of the core language semantics.

In this paper we described the same, together with important resulting properties. This will clarify their usage and provide a sound basis for future models of guidelines and protocols using Asbru.

REFERENCES

[1] AAP: American Academy of Pediatrics, Provisional Committee for Quality Improvement and Subcommittee on Hyperbilirubinemia. Practice parameter: management of hyperbilirubinemia in the healthy term newborn. Pediatrics **94** (1994) 558–565

[2] Peleg, M., Tu, S., Bury, J., Ciccarese, P., Fox, J., Greenes, R., Hall, R., Johnson, P., Jones, N., Kumar, A., Miksch, S., Quaglini, S., Seyfang, A., Shortliffe, E., Stefanelli, M.: Comparing computer-interpretable guideline models: A case-study approach. JAMIA **10**(1) (2003)

[3] Shahar, Y., Miksch, S., Johnson, P.: The Asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines. Artificial Intelligence in Medicine **14** (1998) 29–51

[4] Seyfang, A., Kosara, R., Miksch, S.: Asbru 7.3 reference manual. Technical report, Vienna University of Technology (2002)

[5] Balser, M., Duelli, C., Reif, W.: Formal semantics of Asbru – an overview. In: Proceedings of IDPT 2002, Society for Design and Process Science (2002)

[6] A. Benveniste, G.B.: The synchronous approach to reactive and real-time systems. In: Another Look at Real-time programming. Proceeding of IEEE (1991)

[7] Balser, M., Reif, W., Schellhorn, G., Stenzel, K., Thums, A.: Formal system development with KIV. In Maibaum, T., ed.: Fundamental Approaches to Software Engineering. Number 1783 in LNCS, Springer-Verlag (2000)

[8] Terenziani, P., Carlini, C., Montani, S.: Towards a comprehensive treatment of temporal constraints in clinical guidelines. In: Proc TIME 2002. (2002) 20–27

[9] Votruba, P., Seyfang, A., Paesold, M., Miksch, S.: Environment-driven skeletal plan execution for the medical domain. In: Proc ECAI 2006. (2006)

[10] Dechter, R., Meiri, L., Pearl, J.: Temporal constraint networks. Artificial Intelligence **49**(1-3) (1991) 61–95

[11] Rit, J.F.: Propagating temporal constraints for scheduling. In: Proc AAAI 1986), Morgan Kaufmann (1986) 383–388