A Control Flow Prototype for a Dose Recommending Device for Chronic Myeloid Leukemia Patients

Alena Simalatsar[1], Wenqi You[1], Dechao Sun[1], Verena Gotta[32], Nicolas Widmer[3], Giovanni De Micheli[1]

[1] École Polytechnique Fédérale de Lausanne, Switzerland
[2] School of Pharmaceutical Sciences, University of Lausanne and Geneva, Switzerland
[3] Division of Clinical Pharmacology, Centre Hospitalier Universitaire Vaudois and University of Lausanne, Switzerland

ABSTRACT

In this paper we present a prototype of a control flow for an *a posteriori* drug dose adaptation for Chronic Myelogenous Leukemia (CML) patients. The control flow is modeled using Timed Automata extended with Tasks (TAT) model. The feedback loop of the control flow includes the decision-making process for drug dose adaptation. This is based on the outputs of the body response model represented by the Support Vector Machine (SVM) algorithm for drug concentration prediction. The decision is further checked for conformity with the dose level rules of a medical guideline. We also have developed an automatic code synthesizer for the *icycom* platform as an extension of the TIMES tool.

Keywords

Control flow, personalization, medical guidelines, code synthesis

1. INTRODUCTION

The patient treatment phase of a medical procedure, once the diagnosis has been performed and a specific treatment protocol has been chosen, has usually a long duration. For example it can last for years for some chronic diseases, like Chronic Myeloid Leukemia. Therefore, patients can benefit from the automation of treatment process, which may both improve therapy effectiveness and lifestyle quality. Moreover, the average quality of patients treatment would benefit a lot from the process personalization and adaptation to patients' specific parameters that may widely vary. The behavior of drug dispensing devices is defined by the software running on them which may involve a personalized decision-making process based on the evaluation of specific patient conditions. A successful example of the treatment process automation are the various models of *insulin* infusion pump developed to ease the lifestyle of diabetic patients. The drug dose adjustment procedure for Chronic Myeloid Leukemia patients is different and requires monitoring of other patient parameters. The therapeutic range of the drugs used to treat dynamically progressing Leukemia is narrow and, therefore, there is a very high risk of patients over- or under-dosing, which can cause various adverse events or have no positive effect on patient state respectively. Therefore, the control flow with a feedback of a potential infusion pump for Chronic Myeloid Leukemia patients is an absolutely new design task with specific requirements.

When treating patients medical doctors are often following medical Guidelines (GLs) that are also called clinical protocols. The treatment description includes not only the action-based step-by-step procedures but also recommendations to the action changes upon the definition-based response to the treatment also specified by GLs. Medical GLs synchronize the processes of medical data acquisition, decision-making and treatment provision. Therefore, they can be used to build a control flow with a feedback loop for an autonomous embedded device that also provide adaptation based on the current patient condition. Due to informal representation, medical GLs often suffer from structural problems that introduce additional difficulties to the GLs implementation as a control flow. In our previous works [18,19] we have proposed to use Timed Automaton extended with Tasks (TAT) [7] model, an extension of the Timed Automaton (TA) [6], for medical GLs formal representation and validation of the structural properties.

Imatinib is a drug used to treat patients with newly diagnosed Philadelphia positive (Ph+) Chronic Myelogenous Leukemia (CML), Gastrointestinal Stromal Tumors (GISTs) and a number of other malignancies. In this paper we present a prototype of an *a posteriori imatinib* dose adaptation control flow for a closed-loop autonomous device. The control flow can be then analyzed in TIMES tool [1] and synthesized into the executable C code for *icycom* [2] embedded platform. The feedback loop decision-making process of the control flow is performed based on the personalized prediction of drug concentration values in the blood computed using the Support Vector Machine (SVM) algorithm [27, 28]. The parametrized version of the SVM algorithm allows modeling of the complete concentration-time curve. This way we are able to calibrate the curve every time when a new real drug concentration measurement is given. We can also take into consideration the effect of the residual drug concentration after previous intakes with is essential for a device performing continues dose adjustment.

The paper is organized as follows. Section 2 presents some related work and summarizes the medical GLs formalization approach. We also give an example of modeling the *imatinib* dose adjustment part of the protocol using TAT. Section 3 presents a general control flow and the details of the feedback loop realization, while subsection 3.1 describes the parametrized SVM algorithm used as a body reaction model. In Section 4 we introduce the actual control flow. Section 5 discusses the code synthesis aspect of the control flow. In Section 6 we discuss the perspective evalution of the presented work. Section 7 concludes the paper.

2. BACKGROUND INFORMATION

There have been many frameworks and languages [10, 13, 20, 23, 24] developed in the past years in order to assist medical doctors as well as patients. Many of them such as PRODIGY [13], EON [24], GLIF3 [10], PROforma [20], SAGE [23] represent a class of tools used to build complex decision-support systems. They adapt flowcharts as a core formalism to represent a sequence of actions that are supported by ontology based medical terminology interpretation modules. Most of these tools also provide links to patients' databases. However, they enable validation of GLs structure only by means of their formal representation and have no support for the automatic verification of their formal properties. There exist frameworks such as GLARE [22] and Asbru [11] that provide translation links to model checking environments such as SPIN [3] and SMV [4]. However, if a verified property fails it is difficult to trace back the result needed to change the initial protocol model. Moreover, these formalisms provide the notion of time only in terms of actions order and association of time periods with respect to the patient condition evaluation, which enables only the validation of GLs structure. While, it is important to be able to map medical actions into the time scale in order to verify timing properties of a GL and thus close the gap of medical software and hardware interoperability. In [19] we have used the Timed Automata extended with Tasks (TAT) models of computation, a well-known approach for modeling the behavior of a real-time system, for medical GLs interpretation.

As an example, here we present a TAT model of a small part of the well known dose adjustment medical guideline for an adult patient of the drug called *imatinib*, marketed by Novartis as Gleevec or Glivec. It is a drug used to treat Chronic Myelogenous Leukemia (CML), Gastrointestinal Stromal Tumors (GISTs) and a number of other malignancies. A complete text of the *imatinib* drug administration protocol can be found here [5].

The model depicted on Figure 1 includes only a small part of the *imatinib* GL describing the dose adjustment model for an adult patients with newly diagnosed Ph+ Chronic Myeloid Leukemia (CML) for whom bone marrow transplantation is not considered as the first line of treatment. A more complete model can be found in [18].

The dose should be administered once a day. Medical tests of the level of neutrophils (*N_N*) and platelets (*N_T*) are usually performed every 2 weeks. The dose adjustment model uses the latest data of the medical tests and adjust the dose and intake interval of the drug. The recommended dose of *imatinib* is 400 mg/day for patients in the chronic phase (transition ($\hookrightarrow$) from *Init* to *chronic_p*) of CML and 600 mg/day for patients in the *accelerated* phase (*Init* $\hookrightarrow$ *blast_accel*) of CML. Therefore, the first two transitions of the model represent the choice of the treatment according to the patient condition.

The dose may be increased from 400 mg to 600 mg in patients with the *chronic* phase of the disease (*chronic_p* $\hookrightarrow$ *LoL_resp*) or from 600 mg to a maximum of 800 mg given as 400 mg twice daily (*blast_accel* $\hookrightarrow$ *LoL_resp* and $p1 = p2$, where $p2 = half\ day$) in

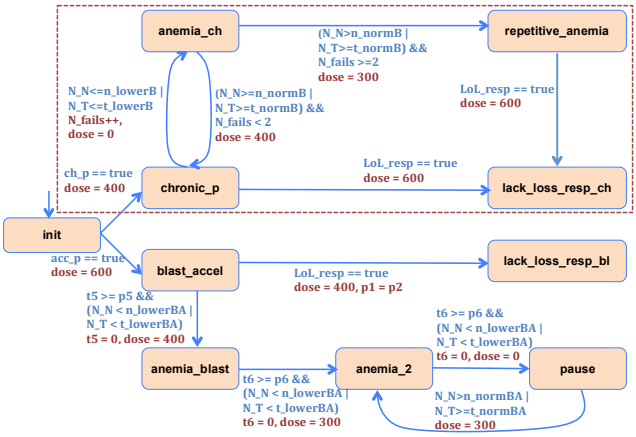

Figure 1: Drug delivery, measurements and *imatinib* dose adjustment models

patients with accelerated phase or blast crisis in case of: *(i)* disease progression at any time; *(ii)* failure to achieve a satisfactory hematological or cytogenetic response; or *(iii)* loss of a previously achieved hematological and/or cytogenetic response.

In the *chronic* phase of CML, marked with a rectangle in Figure 1, if the level of neutrophils (ANC) goes below 1.0 x 10^9/l (*N_N<=n_lowerB*) and/or level of platelets goes below 50 x 10^9/l (*N_T<=t_lowerB*), then the following actions can be taken:

1. Stop *imatinib* until ANC >1.5 x 10^9/l and platelets >75 x 10^9/l (*chronic_p* $\hookrightarrow$ *anemia_ch*);
2. Resume treatment with *imatinib* at previous dose, (*anemia_ch* $\hookrightarrow$ *chronic_p*, *N_fails* accounts to the number of anemia occurrences);
3. In the event of recurrence of ANC <1.0 x 10^9/l and/or platelets <50 x 10^9/l, repeat step 1 and resume *imatinib* at reduced dose of 300 mg (*anemia_ch* $\hookrightarrow$ *repetitive_anemia*);

The treatment of a patient in the *accelerated* phase of CML or in the *blast crisis* (starting dose 600 mg) is represented in the lower part of Figure 1.

Formal representation of a medical GL is understandable by a machine. Thanks to the GL representation with TAT we can perform its structural validation as was presented in [18] and fix certain problem. When being combined with the technical requirements for an embedded system it can be used to build a medical guideline oriented system aimed to assist medical doctors in patients treatment. From the system design point of view a patient is a reactive system that responds to any treatment events. This way we need to approach the task of medical systems design as of a cyber-physical system where the actions of the cyber part are changing with respect to the state of a physical system.

In [14] authors present a formal approach to the development of a Generic Patient Controlled Analgesic (GPCA) infusion pump. They approach the problem of the safety-assured development of the pump software by using TA model. From the point of view of an embedded system design we take a similar approach to drug dose administration support system modeling. However, our system has different requirements and principals. Their system feedback loop monitors the parameters that indicate the event of patient over-dosing (going above certain boundary). In our case we guard two boundary conditions that may indicate over- or under-dosing of

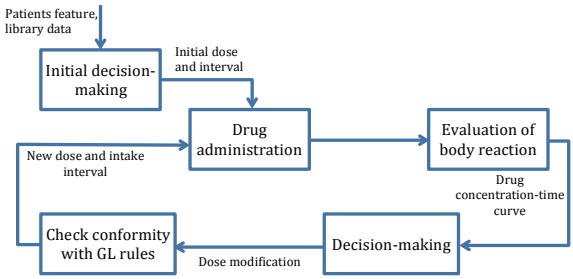

Figure 2: General feedback loop control flow

a patient. Moreover, the monitored parameters are different. What is more important, the feedback-loop of the control flow presented in Section 4 includes an algorithm able to predict the patient state in a personalized manner. The use of the algorithm allows to prevent adverse events of medication.

3. FEEDBACK LOOP REALIZATION

Figure 2 depicts our representation of a general control flow with the feedback loop for an autonomous drug administration support device. It starts with an initial decision-making process that provides the initial dose and intake interval. The *drug administration* block can either perform the actual drug delivery or remind a patient or a caregiver when the drug needs to be administrated and with what dose for oral drugs such as *imatinib* [5]. Once the drug is administered, it is essential to evaluate the effect of the treatment reflected in the *evaluation of body reaction* block of the diagram. The decision regarding the next dose is then take based on the body reaction (the *decision-making* block) and must conform with the medical GL for this specific drug administration (*check conformity with GL rules*). This way we have a cyclic procedure with a feedback loop. Further in the text we will refer to the design of a control flow for an embedded device that can either perform the automatic drug delivery, or remind a medical doctor or a patient when a specific dose need to be administered/taken.

The realization of a feedback loop for a medical system requires a patient body reaction model, that can provide us with the information about the response to the treatment required by the medical GL. The GL model in turn should be able to react according to the new data. Unfortunately, the GL model presented above requires a body model that can return the level of White Blood Cells (WBC), e.g., neutrophils and platelets, which is hard to build. Moreover, even if such patients model exists, the GL model presented above does not account for other patient features and thus does not provide personalization.

Various clinical studies show that responsiveness to the treatment with drugs depends on the concentration of the drug in patient's blood that depends on patients features, drug dose and intake interval. Pharmacokinetics (PK) is a branch of pharmacology focused on studying the drug disposition in the human body. For lots of drugs the concentration in patient's blood is highly related to its effects. Pharmacodynamics (PD) is the study of the biochemical and physiological effects of drugs on the body. Therapeutic Drug Monitoring (TDM) [15] is the approach that unifies the PK-PD knowledge. The concentration in the patient's blood may be closely related to the drug effect (PK-PD relationship). Drugs with clear PK-PD relationships and a narrow therapeutic range may be easily under- or overdosed. The intervention of TDM and dosage individualization can help keeping the drug concentration within the limited ranges provided by the PD studies. The real TDM measurement procedure is quite slow (takes about a day), expensive and requires an invasive measurement of individual drug concentration. Currently, when treating Chronic Myeloid Leukemia TDM is performed only in case of adverse events or suboptimal response in order to figure out whether it is related to over- or under dosing or not. This way, when applying TDM, an algorithm that is able to compute the drug concentration in patients blood at certain point, would not only represent a surrogate of body reaction and thus close the loop of the control flow but also lower the costs of the real TDM.

Recently, the SVM-based algorithm has shown a great potential in drug concentration prediction [26–28] without requiring a measurement of individual drug concentration. This algorithm is trained with data (a library) that include real measured values of *imatinib* concentration obtained trough routine TDM with corresponding patients parameters. The collection of such library data is time consuming and thus expensive. However, once the data are available, they can be used by the SVM algorithm with no additional cost. This way it opens new possibility in patient treatment procedures. However, in case of Chronic Myeloid Leukemia the TDM can not be used as a standalone approach should not contradict with the existing approved GL presented in Section 2. According to [8] the response evaluation is performed every 3 months. The formal model of response evaluation is presented in [18]. The check for quantitative adverse events is usually performed every 2 weeks, especially at the beginning of the treatment.

When applying the SVM algorithm each following drug concentration must account for the residual concentration in the blood. Therefore, we need to keep track of the concentration values after each dose intake and thus performe SVM computation every day even though the drug concentration may not undergo drastic changes in one day. The more frequent application of the TDM or its replacement with an algorithm, can improve the quality of treatment by preventing adverse events and lack of response caused by the suboptimal dosage. It is also possible to appropriately adjust the dose when the previous intake was missed out. The dose adjustment can be finer (e.g. 50mg) compared to the one of the approved *imatinib* GL (100mg). We only need to make sure that the dose and intake interval values adapted according to the algorithmic computation corrected from time-to-time with a real measurement lay within the ranges allowed by the approved GL.

This way we can extract the following rules that need to be checked in the *check conformity with GL rules* block of Figure 2:

1. $dose >= 300$ - the minimum dose assigned should be not less then 300mg;
2. $dose <= 800$ - the maximal dose assigned should be not more then 800mg;
3. $dose >= 800 \rightarrow p:=p/2,\ dose:=dose/2$ - when the dose is equal (or greater) than 800mg it should be administrated in two shots.

3.1 Parameterized Support Vector Machine Algorithm

In this section we present a patient body (physical) model that is able to predict the drug concentration values over time required by the TDM approach to modify the behavior of the cyber part of our medical system. The Support Vector Machine(SVM) model able to predict the drug concentration in the blood was initially developed to support medical doctors in the clinical routine. There, in the simplest case, once a new patient comes, an initial dose is determined based on a population value estimated from the library data.

This first dose computation is called an *a priori* drug dose adaptation. However, the real measurements of the drug concentration need to take place in order to make sure that drug concentration is within the therapeutic range defined by the PD studies. If it is too high or too low the dose needs to be decreased or increased, respectively, which is known as an *a posteriori* drug dose adaptation. The population based dose prescription does not guarantee that the new patient and the library datasets have similar conditions. The SVM-based first dose estimation, in turn, is able to take into consideration the patient's features and then predicts the drug concentrations at a specific time or the concentration curves from which a dose can be derived.

The initial Support Vector Machine(SVM) algorithm applied to do the point-wise prediction of the drug concentrations in patients blood, has been already presented in our previous works [27, 28]. Later it was enhanced with the RANSAC algorithm [26] to increase the precision of the prediction. Though, the visualization of the concentration-time curve is important for the PK-PD studies. An analytical formula that would capture the shape information of the curve is essential when we need to calibrate the curve, as needed in the *a posteriori* drug dose adaptation. Newly measured concentration values or the residual concentration of the drug in the blood after previous drug intakes (e.g. at 24 hours) can be used for curve calibration and accounting for consequent multiple doses. Parameterized Support Vector Machine (ParaSVM) uses the curve *parametrized* database build using the RANSAC algorithm as in [26] as an input. RANSAC is an algorithm to separate inliers and outliers from a set of (noisy) data with respect to the given basis functions. In short, it randomly selects a very small subset of the given input data, computes the weights of each basis function considering the small subset, and then determines the inliers and outliers for the rest of the data with a given distant value (threshold). Here, instead of separating inliers and outliers, RANSAC algorithm is first applied to compute the basis functions of the concentration-time curve. Then it is also applied to compute the weights of the basis functions for each patient that form the Parameter Library used as the training data.

3.1.1 SVM-based Parametrization (ParaSVM)

In case of modeling N patient samples, the form of patient samples becomes $(x_i, y_i^1, \cdots, y_i^j, \cdots, y_i^{NP})$, where i is the ID of a sample $i \in \{1, 2 \cdots, N\}$, y_i^j denotes the j-th parameter value of this patient, and NP is the number of parameters. The goal is to find NP linear functions $f^j(x) = w^j \cdot \phi^j(x) + b_j$ to describe the relationship between the dataset points and estimate the parameter value y according to a new input dataset. For that we need to minimize the following modified objective function:

$$\min_{w,b} \frac{1}{2}||w||^2 + C_0 \sum_{j=1}^{NP} \sum_{i=1}^{N} [y_i^j - w^j \cdot \phi^j(x_i) - b^j]^2 \qquad (1)$$

Applying Lagrangian analysis to solve the optimization problem of objective function, we obtain w as:

$$w^j = \sum_{i=1}^{N} \alpha_i^j \phi^j(x_i) \qquad (2)$$

Combining Equation (1) and (2), we can obtain a linear system to:

$$\begin{bmatrix} \mathbf{K}^j + \frac{1}{C_0} I & 1 \\ 1^T & 0 \end{bmatrix} \begin{bmatrix} \alpha^j \\ b^j \end{bmatrix} = \begin{bmatrix} y^j \\ 0 \end{bmatrix} \qquad (3)$$

where each entry of the kernel matrix $\mathbf{K}^j$ is defined to be $K_{ab}^j = \phi^j(x_a)^T \phi^j(x_b)$. Therefore, the prediction function becomes: $f^j(x) = \sum_{i=1}^{N} \alpha_i \mathbf{K}^j(x_i, x) + b^j$.

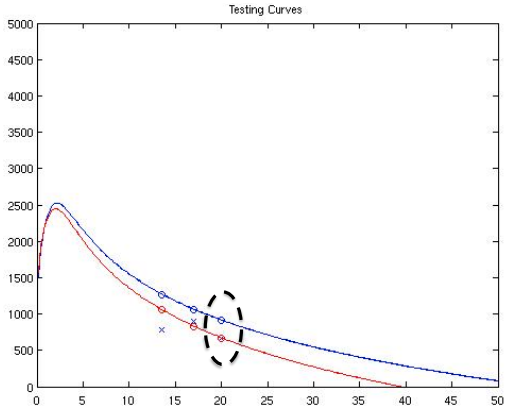

Figure 3: An examples of parametrically refined drug Concentration to Time Curve (dCT). Upper curve: dCT curve computed by ParaSVM; lower (curve: dCT curve adjusted with one measurement; cross: new real measured concentrations; circle: predicted concentration based on the dCT curve. Dotted black circle: measured point chosen for curve adjustment. X-axis: time [h], Y-axis: concentration value [$\mu g/L$]

.

In the *a posteriori* dose adaptation, we refine the predicted drug concentration curve computed by ParaSVM. First of all we still need to compute the initial concentration curve, the one that starts with a 0 residual drug concentration. Therefore, we set the following conditions to obtain the first refined curve:

- Starting point of the concentration curve should be less than or close to 0:

$$\lim_{t \to 0} f_{concentration}(t) \leq 0; \qquad (4)$$

- Ending point of the concentration curve should be less than or close to 0:

$$\lim_{t \to T} f_{concentration}(t) \leq 0 \qquad (5)$$

  where T is determined as the time when the concentration curve drops below 0, i.e. $T = 72h$ (at least 3 days);

- After giving the dose, the concentration value should start growing with time:

$$\frac{\partial f_{concentration}}{\partial t}|_{t=0} > 0; \qquad (6)$$

- After several hours, the concentration value reaches the peak value and starts to decrease:

$$\frac{\partial f_{concentration}}{\partial t}|_{t=T_p} < 0, \qquad (7)$$

  where T_p is a time point after the peak value, i.e. we set it as $T_p = 24$, since *imatinib* need to be administered once a day.

- The concentration curve whose shape is the most similar compared with the one predicted from ParaSVM will be chosen:

$$\min_{y_r} \sum_{j=0,\cdots,NP} (y_r^j - y^j)^2 \qquad (8)$$

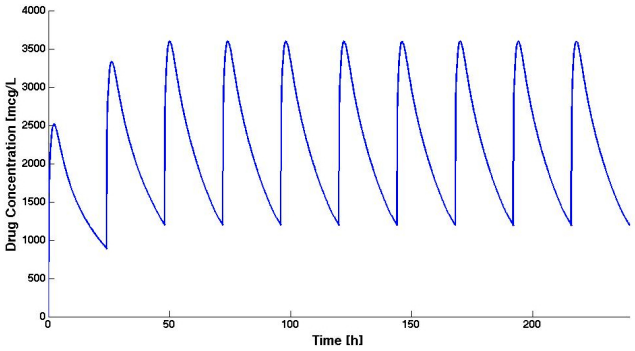

Figure 4: An example of multi-dose estimation of the dCT curves for 10 days.

where y^j stands for the predicted parameters using ParaSVM for the concentration curve and y_r^j denotes the parameters for the refined curve.

Figure (3) shows an example of parametrically refined drug concentration curves. The upper curve on the figure indicates the predicted concentration curve without refinement while the lower one is the refined concentration curve according to the rules listed above. Crosses stand for the newly-measured concentration values and circles are the predicted concentrations with respect to the measuring time. One measured value is randomly picked up for the refinement which is indicated by a dotted black circle. Therefore, in the dotted black circle, the cross value should appear in the same position as the red circle, and the accuracy improvement can be calculated from the remaining points. In Figure 3 the refined concentration curve improves the prediction accuracy for the other two measurements with respect to the measurement in the dotted circle.

3.1.2 Multiple Dose Analysis

Knowing how the concentration varies with respect to multiple doses is essential for a continues drug dose adapting procedure. As it takes usually more than 24 hours for a drug to be cleared out from the human body, we have to consider the residual concentration values from the previous dose(s). Such curve adjustment, as presented above, can be used to account for this. Figure 4 shows an example of estimating the drug concentration over 10 days based on ParaSVM taking into account the residual drug concentrations. In an *a posteriori* case, after we adjust the concentration curve with a given measurement, the multi-dose concentration-time curve can be obtained. From this curve, it is visually easy to obtain the peak and trough concentration values and to check whether they are within the therapeutic ranges or not. From Figure 4 we can easily see, that the residual concentration affects both the peak and trough concentration values. We can also see that if patients parameters do not change we can enter into a steady state of drug concentration. However, this steady state can be easily disturbed by such external events as patient parameters changes, miss of a dose or complementary medication.

4. THE CONTROL FLOW

In the previous sections we have briefly described an approach to formal representation of a medical GL showcasing it with an example of modeling a small part of the well-known *imatinib* dose adjustment guideline (see Section 2). In Section 3 we have described our approach to the feedback loop realization, where the body reaction model is represented by the Parametric SVM algorithm described in details in Section 3.1. In this section we present an executable control flow that refines the one presented in Section 3 combining body reaction model, GL rules and results of the pharmacodynamics (PD) studies.

The control flow that can be executed on an embedded electronic device is presented in Figure 5. Following the general feedback loop control flow of Figure 2 it is divided into *initial decision-making*, *drug delivery*, *body reaction*, *decision making* and *check for conformity with GL rules* blocks. The flow has fixed values for the intake intervals: one day p_2 or half a day p_3; while the dose can be changed with more fine than in currently applied *imatinib* GL values defined by the parameter Δ_{dose} (e.g., 50 mg).

The flow starts with the initialization of the peripheral of an embedded device. Here the daily dose is also initialized with a personalized values (*d_dose:=init_dose*) after being computed externally using the SVM-based algorithm taking into account all the patient features. The period p is first set to one day ($p:=p_2$) in case the initial dose value is less then 800 mg, while the actual dose to be administrated is set to the daily dose (*dose:=d_dose*).

The *drug delivery* block is composed of *main* and *deliver* blocks. Here we will stay in the *main* block until clock T_2 is less then intake period p. Once the condition $T_2 >= p$ holds we transit to the *deliver* block, where either the drug is automatically administrated, in case of an implantable device or a reminder is given to a user. Immediately after each drug administration the body reaction must be evaluated where SVM-based prediction of the drug concentration performed in the SVM-TDM block. The algorithmic computation of the body reaction is performed every time after the drug administration (every day or half a day), while the clinical measurement of the drug concentration is performed (every period p_1, e.g. every 2 weeks). The value of the real measurement is further used to correct the SVM-TDM algorithm.

In the *decision-making* block the decision about increasing, decreasing or keeping the dose is taken. In the present example we assume that the trough concentration value C_{min} (at 24 or 12 hours after the last intake) should lay within the 750:1500 $\mu g/l$ range ($750\ \mu g/l <= C_{min} <= 1500\ \mu g/l$) [12, 25]. If C_{min}<*750* $\mu g/l$ the daily dose will be increased (*d_dose* += Δ_{dose}) and decreased in case C_{min}>*1500* $\mu g/l$.

In the next block we check if the value of the daily dose and intake interval are conformed with the rules defined in Section 3. For example, when *d_dose*>*800 mg* the period must be set to half day interval and the dose divided by 2 for each drug administration. The alarm (*alarm1!*) will be also generated since the maximal dose defined by *imatinib* GL is exceeded. The period will be set back to one day next time only after the *d_dose* falls down to 600 mg. An alarm (*alarm2!*) will be generated when we reach the minimal dose value defined in *imatinib* GL.

5. IMPLEMENTATION

The model presented above is directly implementable with TAT in TIMES toolbox. There might be several variations of the implementation mainly concerning the small operations with intake interval and dose values when the dose must be administrated in 2 shots. They can be implemented either as functional tasks associated with corresponding locations, or as assignments of the corresponding transitions. The SVM-TDM algorithm is quite complicated and must be modeled as a locations associated task. The control flow represented with TAT is synthesizable either into an executable code for an embedded platform or into a decision-support tool. When implementing the control flow as an embedded software it is essential to perform schedulability analysis, which check that none of the task deadlines will be actually missed, considering

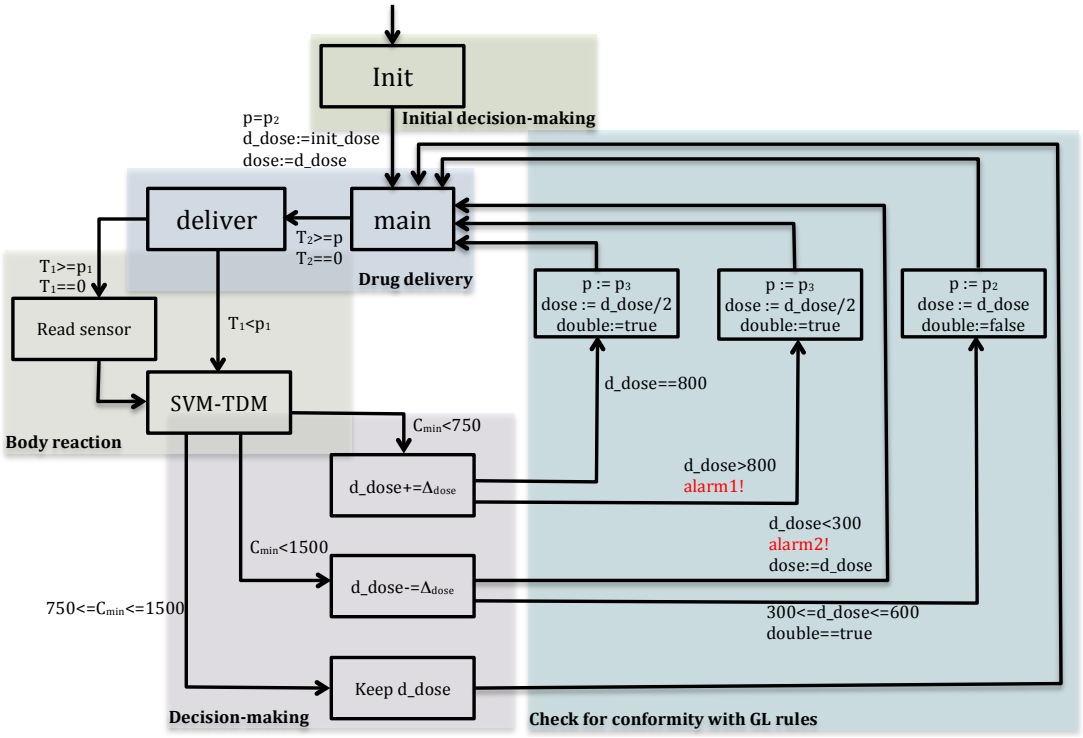

Figure 5: Synthesized control flow

the chosen scheduling policy and tasks parameters, such as worst-case execution time.

TIMES toolbox implements TAT model and has already provided a platform independent code synthesizer, that is able to generate parts of the code that do not include any functions specific to an embedded platform. Among the most essential platform dependent code parts we can name the automatically generated *Makefile*, initialization of the platform peripherals, and, the most important, operations with clocks. We have extended the existing TIMES code synthesizer in order to be able to generate the executable code for *icycom* platform [2]. The code generator is based on platform independent code generator of TIMES with several features added. Further we describe some details of the new code generator implementation.

5.1 System Initializing and the Initial Task

Each embedded platform needs to be initialized at the beginning. Therefore, we have added an obligatory *Init* location with an associated initialization task that can be modified and adapted for any target platform independently from the control flow. In the generated code, the platform initialization part is completed at the beginning of the main function, in the SYSTEM_START() procedure. The *Init* function is present in our control flow depicted on Figure 5. It has to run before all other tasks and be executed only once. In this task, the libraries and the initializing procedures of the optional peripheries are included. Besides that, the MACRO functions defined in this task can be used in all other task. Based on the periphery libraries in this task, the corresponding symbols will be added to the *Makefile* automatically.

5.2 The Global Variables Passing

The synchronization using global variables is not a trivial task. We had to rework the implementation of the global variables passing in the new code generator, since we often use it in our case studies, e.g. when modifying the dose and intake interval values. The global variables defined in TAT can be evaluated and set in the guard and assignments of the automata network, as well as in the task body. If a global variable is used in a task, it has to be added to the *input* of the task body editor in TIMES. The value of the global variable in this case is copied to a local variable when the task is added to the queue. During the task running, the local variable is used in the task body, rather than the global one. After the task body is finished, the value of the local variable will be copied back to the global one. That is to say, the local value used in a task body equals to the global value when the task is added to the queue. The transient value of local variable during the task body execution will not affect the global one. The value of the global variable will be changed by the task body only when the task body is finished.

If the maximum number of the equal tasks in the queue is greater than 1, then we need to use more local variables. The local variables for the same task work like a FIFO. The task body first added to the queue takes the first local variable. If the second one is added to the queue before the first task of the same type is finishes, the current, not yet modified, value of the global variable will be stored at the second position of local variables FIFO. The third task will store the current values of the global variable at the third position and so on. If the first task finishes, the value of the first local variable will be copied to the global one and the value in the second position of the local variables FIFO will be copied to the first one, the value of the third one will be copied to the second one, etc. The time diagram for global variable passing is presented in Figure 6. Let's assume that we have a model with two tasks except the *init* one and a global variable *dose*. One task named *dose_adj* adjusts the value of the global variable, and the other task named *dose_deliver* delivers the drug based on the value of the global variable. The maximum number of the *dose_deliver* task in the task queue is two and the maximum number of the *dose_adj* task is one. The behavior of tasks is shown in the upper graph while the

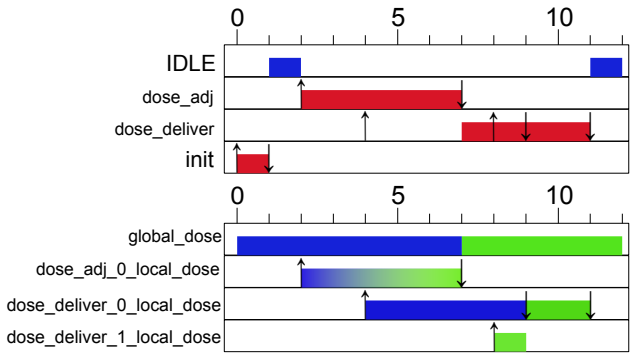

Figure 6: A example for the global variable handling

changes of the global and local variables are shown in the lower one in the Figure 6. We can see the execution of the *Init* function before any task. Then the *dose_adj* task is added to the queue at the 2nd cycle and keeps running for 5 cycles. When it is added to the task queue, the value of the *global_dose* is copied to the local variable, *dose_adj_0_local_dose*. The local variable changes during the task *dose_adj* task execution. After the *dose_adj* task finishes, the value of *dose_adj_0_local_dose* is copied to the global one, so the *global_dose* is changed at the 7th cycle. The *dose_deliver* task is added to the queue in the 4th cycle. The global variable copied to the *dose_deliver_0_local_dose* variable still has the unchanged initial one. Thus, the first *dose_deliver* task running during the 8th and 9th cycle is based on the initial value of *dose*. The second *dose_deliver* task is added to the task queue in the 8th cycle and the updated global variable is copied to the *dose_deliver_1_local_dose* variable. When the first *dose_deliver* task is finished, the value of the *dose_deliver_1_local_dose* variable is copied to the *dose_deliver_0_local_dose* variable and further used by the second *dose_deliver* task. This way of handling global variables passing is valid only for the First Come First Serve scheduling policy. For other scheduling policy, since the order of tasks execution might change dynamically, the scheme will be more complicated. It is essential to be aware of the global variables passing mechanism when building new models.

6. DISCUSSION

Using TAT for modeling medical GLs as well as a control flow is not a unique approach. However, TAT has several advantages. First of all it has a notion of time and allows us modeling of the choice of sequential actions based on specific conditions mapped to the time scale. It enables the modeling of several plans with periodically repeating actions that can be executed in parallel with other actions. It can associate tasks with locations to model simple actions (e.g., computational blocks or medical tests) that need to be finished before certain deadline. When the model is executed a task will be added to the scheduling queue whenever an associated location is reached. The tasks will be then executed in a scheduler defined order. TAT is *expressive* enough to represent the step-by-step actions of medical GLs.

Furthermore, it is able to assist a designer in correcting structural problems of the GLs such as *incompletenesses*, *inconsistencies*, *ambiguity* and *redundancies* and provides the *verification* abilities of the methodology. TIMES [1], the tool that implements TAT, includes a model-checker engine that supports verification of system properties. Therefore, when the protocol is formally represented with TAT we are able to perform an automatic verification of the protocol structural properties, such as the reachability/non reachability of some states, or to be able to find a path that would avoid certain actions, e.g. surgery or chemotherapy. A big variety of properties to be verified can be found in [9, 16, 21]. As was already described in Section 5 a medical protocol represented using TAT can be turned into a fully deterministic model [7] thus enabling the code synthesis in two different directions: *(i)* to produce a personalized decision-support tool adapted for a patient conditions similar to the idea of [17] or *(ii)* to do the code synthesis for an executable code an embedded system as in [14]. The main control flow of the electronic devices as well as a complementary decision-support system should be based on parts of the same medical GL.

When synthesizing a decision-support tool we can create a framework that would assist a doctor in suggesting the steps in time that need to be taken in general patient treatment procedure. Patient condition include not only the parameters of his/her state but also the combination of diseases and therefore treatment procedures with their effects that may interfere, and thus create other complications, not related to the initial disease. Patient treatment procedure often includes the combination of more than one standard medical treatment GL. Therefore, it is important to be able to synthesize a patient-oriented personalized decision-support tool from a complex model of cooperating formal representations of the GLs that may interfere with each other.

The control flow presented in this paper can be extended to add more features. It can not only perform the computation of specific values (dose or drug administration period), but also communicate with an external server that performs this computation in order to receive the updated values, trigger the operational code that reads and analyzes data coming from the sensor, activate the actuators of the chip or send recommending alarms.

7. CONCLUSION

In this paper we have presented prototype of the *imatinib* dose adjustment control flow for the personalized a posteriori drug dose adaptation using Timed Automata extended with Tasks(TAT). When implemented in TIMES the control flow is synthesizable into an executable code for the *icycom* embedded platform using our extended platform independent code generator. The feedback loop of the control flow includes the parametrized Support Vector Machine (SVM) algorithm based model of the body reaction that is able to predict the drug concentration over time curve in the patients blood in a personalized manner. The drug concentration value is compared with the therapeutic range defined by the Pharmacodynamics (PD) studies of *imatinib* and decision about the modification of the drug dose (increase or decrease) and the intake time interval for a new patient with certain parameters is then taken. The dose value is then checked for conformity with the dose level rules of the medical GL. We also show that the control flow conforms with the currently approved *imatinib* protocol. Such concept will have to be validated with real clinical data.

8. ACKNOWLEDGMENTS

This work was supported by ISyPeM project of the Swiss NanoTera initiative, evaluated by the Swiss National Science Foundation. The authors would like to thank T. Buclin from CHUV Hospital of Lausanne for the discussion regrading the project.

9. REFERENCES

[1] TIMES. http://www.timestool.com/l.

[2] IcyCom platform. hhttp://www.csem.ch/docs/Show.aspx?id=12228.

[3] SPIN. http://spinroot.com/spin/whatispin.html.

[4] SMV. http://www.cs.cmu.edu/ modelcheck/smv.html.

[5] Imatinib. http://www.glivec.com/files/Glivec-400mg-coated.pdf.

[6] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[7] T. Amnell, E. Fersman, P. Pettersson, H. Sun, and W. Yi. Code synthesis for timed automata. *Nord. J. Comput.*, 9(4):269–300, 2002.

[8] M. Baccarani, F. Castagnetti, G. Gugliotta, F. Palandri, and S. Soverini. Response definitions and european leukemianet management recommendations. *Best Pract Res Clin Haematol*, 22(3):331–41, 2009.

[9] A. Bottrighi, L. Giordano, G. Molino, S. Montani, P. Terenziani, and M. Torchio. Adopting model checking techniques for clinical guidelines verification. *Artif. Intell. Med.*, 48:1–19, January 2010.

[10] A. Boxwala, M. Peleg, S. Tu, O. Ogunyemi, Q. Zeng, D. Wang, V. Patel, R. Greenes, and E. Shortliffe. GLIF3: a representation format for sharable computer-interpretable clinical practice guidelines. *Journal of Biomedical Informatics*, 37(3):147–161, June 2004.

[11] G. Duftschmid, S. Miksch, Y. Shahar, and P. Johnson. Multi-level verification of clinical protocols. In *in: Proceedings of the Workshop on Validation Verification of Knowledge-Based Systems (VVÕ98), in conjunction with the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KRÕ98*, pages 4–1, 1998.

[12] V. Gotta, N. Widmer, L. Decosterd, C. Csajka, M. Duchosal, Y. Chalandon, D. Heim, M. Gregor, and T. Buclin. Therapeutic drug monitoring (TDM) of imatinib: Effectiveness of bayesian dose adjustment for CML patients enrolled in the imatinib concentration monitoring (I-COME) study. Swiss Medical Forum, page 285, 2010.

[13] P. D. Johnson, S. Tu, N. Booth, B. Sugden, and I. N. Purves. Using scenarios in chronic disease management guidelines for primary care. In *in Proceedings of American Medical Informatics Association(AMIA) Symposium*, pages 389–93. PubMed Central PMCID, 2000.

[14] B. Kim, A. Ayoub, O. Sokolsky, I. Lee, P. Jones, Y. Zhang, and R. Jetley. Safety-assured development of the gpca infusion pump software. In *Proceedings of the ninth ACM international conference on Embedded software*, EMSOFT '11, pages 155–164, New York, NY, USA, 2011. ACM.

[15] J. H. Martin, R. Norris, M. Barras, J. Roberts, R. Morris, M. Doogue, and G. R. D. Jones. Therapeutic monitoring of vancomycin in adult patients: a consensus review of the american society of health-system pharmacists, the infectious diseases society of america, and the society of infectious diseases pharmacists. *American journal of health system pharmacy AJHP official journal of the American Society of HealthSystem Pharmacists*, 31(1):21–24, 2010.

[16] B. Pérez and I. Porres. Authoring and verification of clinical guidelines: A model driven approach. *J. of Biomedical Informatics*, 43:520–536, August 2010.

[17] I. Porres, E. Domfnguez, B. Perez, A. Rodriguez, and M. Zapata. A model driven approach to automate the implementation of clinical guidelines in decision support systems. In *Engineering of Computer Based Systems, 2008. ECBS 2008. 15th Annual IEEE International Conference and Workshop on the*, pages 210 –218, 31 2008-april 4 2008.

[18] A. Simalatsar and G. De Micheli. Medical guidelines reconciling medical software and electronic devices: Imatinib case-study. In *International Conference on BioInformatics and BioEngineering*, November 11-13 2012.

[19] A. Simalatsar and G. De Micheli. TAT-based formal representation of medical guidelines: Imatinib case-study. In *Engineering in Medicine and Biology Society, EMBC, 2012, Annual International Conference of the IEEE*, pages 5078–5081, Aug. 28 -Sept. 1 2012.

[20] D. R. Sutton, P. Taylor, and K. Earle. Evaluation of PROforma as a language for implementing medical guidelines in a practical context. *BMC Medical Informatics and Decision Making*, 6:20+, Apr. 2006.

[21] A. Teije, M. Marcos, M. Balser, J. van Croonenborg, C. Duelli, F. van Harmelen, P. Lucas, S. Miksch, W. Reif, K. Rosenbrand, and A. Seyfang. Improving medical protocols by formal methods. *Artif. Intell. Med.*, 36:193–209, March 2006.

[22] P. Terenziani, S. Montani, A. Bottrighi, M. Torchio, G. Molino, and G. Correndo. The glare approach to clinical guidelines: main features. *Studies in Health Technology and Informatics*, 101, 2004.

[23] S. W. Tu, J. R. Campbell, J. Glasgow, M. A. Nyman, R. McClure, J. McClay, C. Parker, K. M. Hrabak, D. Berg, T. Weida, J. G. Mansfield, M. A. Musen, and R. M. Abarbanel. The SAGE Guideline Model: Achievements and Overview. *Journal of the American Medical Informatics Association*, 14(5):589–598, June 2007.

[24] S. W. Tu and M. A. Musen. Modeling data and knowledge in the EON guideline architecture. *Medinfo*, 10(1):280–284, 2001.

[25] N. Widmer, L. A. Decosterd, C. Csajka, S. Leyvraz, M. A. Duchosal, A. Rosselet, B. Rochat, C. B. Eap, H. Henry, J. Biollaz, and et al. Population pharmacokinetics of imatinib and the role of alpha-acid glycoprotein. *British Journal of Clinical Pharmacology*, 62(1):97–112, 2006.

[26] W. You, A. Simalatsar, and G. D. Micheli. RANSAC-based enhancement in drug concentration prediction using support vector machine. In *Proceedings of the International Workshop on Innovative Simulation for Healthcare (IWISH)*, pages 21–5, Sept. 19-21 2012.

[27] W. You, N. Widmer, and G. De Micheli. Example-based support vector machine for drug concentration analysis. In *Engineering in Medicine and Biology Society,EMBC, 2011 Annual International Conference of the IEEE*, pages 153 –157, 30 2011-sept. 3 2011.

[28] W. You, N. Widmer, and G. De Micheli. Personalized modeling for drug concentration prediction using support vector machine. In *Biomedical Engineering and Informatics (BMEI), 2011 4th International Conference on*, volume 3, pages 1505 –1509, oct. 2011.