UML-STATECHART MODELING TOOL FOR THE VLE SIMULATOR: AN APPLICATION TO A CHRONIC RENAL DIALYSIS UNIT

O. ROUX

FUCaM/ LSM
Chaussée de Binche, 151
B-7000 Mons - Belgique
roux@fucam.ac.be

D. DUVIVIER[1,2], E. RAMAT[1,2]

1 Univ Lille Nord de France,
F-59000 Lille, France
2 ULCO, LISIC, BP719
F-62228 Calais Cedex, France
{Duvivier,ramat}@lisic.univ-littoral.fr

G. QUESNEL,

INRA, UR875 Biométrie et Intelligence Artificielle,
F-31326 Castanet-Tolosan, France,
gauthier.quesnel@toulouse.inra.fr

C. COMBES

EURISE - Faculté des sciences,
Université Jean Monnet
23 rue du Docteur Paul Michelon
F-42023 Saint-Etienne cedex 2
combes@univ-st-etienne.fr

ABSTRACT: *The various reforms of the health care system aim at reducing the corresponding costs. So the hospitals must rationalize their costs while guaranteeing a good quality of service. In order to reach this goal, different scenarios of health care unit management have to be envisaged and carefully evaluated before their implementation on the system. Thus, simulation is used more and more frequently to evaluate the performances related to the reorganization of the health care units. However the elaboration of accurate simulation models is not an easy task. In this paper, we use a modeling tool dedicated to performance evaluation via the VLE (Virtual Laboratory Environment) simulator in the case of a chronic renal dialysis unit reorganization. The modeling part is based on the UML statecharts (Unified Modeling Language) which are used to generate VLE simulation model and XML file for the parameterization*

KEYWORDS: *Modeling tool, UML, statecharts, simulation, VLE, dialysis unit.*

1 INTRODUCTION

Due to various reforms, the nowadays hospitals have to reduce their outlay and improve their reactivity.
Many efforts have to be accomplished to adapt their systems to the patients' needs, the technological evolutions and the social and economic context.
In order to control these systems, many problems have to be solved from the design phase. Due to the investments required from the implementation of a new health care unit, the dimensioning problems are very important. During the exploitation phase, to change the hospital functioning is difficult or even risky. It is necessary to be able to evaluate a priori the functioning and the restructuring of these systems and to anticipate the effects of an important workload increase on their behavior. This short overview of the problems relating to hospitals is sufficient to justify the need of an a priori or a posteriori evaluation of hospital performances faced to a workload. These problems can be solved in using the modeling and performance evaluation techniques (by discrete-event simulation) [Law99]. Simulation is a useful tool to reproduce the functioning of a system and to test several configurations or management rules. But two difficulties appear. The first one consists in analyzing the system in order to build a model. The second one consists in translating the obtained model into a simulation language. These two difficulties have been solved thanks to the elaboration of a methodological framework which allows on the one hand to apprehend the complexity of these systems and to design representative models of the reality, and on the other hand to obtain the corresponding simulation model (program) in an almost semi-automatic way. In the article, we present an application of a modeling tool in the case of a chronic renal dialysis unit. The method includes the collection and formalization of hospital needs, the identification of objectives, the analysis and specification of the entities constituting the system, and the semi-automatic translation into a discrete event simulation language which is in this case, VLE [Quesnel09].
The proposed framework uses UML (Unified Modeling Language) [OMG04] and is inspired by MDA (Model Driven Architecture) [Mukerji01] concept for modeling and specification stage.
The proposed modeling tool is based on generic meta-models described in UML class diagrams. These models are afterwards specialized for a system class and instantiated for a specific case of the system class. The obtained model is translated into a simulation language in using MDA methodology. MDA is a methodology of development proposed by the OMG (Object Management Group)[1]. It allows to separate the functional speci-

[1] http://www.omg.org

fications (from a system of the specifications) to its implementation on a given platform. In this purpose, MDA defines architecture of specifications structured in Platform Independent Models (PIM) and in Platform Specific Models (PSM). It proposes a modeling approach based on model patterns like the design patterns of the object-oriented paradigm. It needs to develop translator to convert UML model into PSM in our case, a translation in a discrete event simulation language such as VLE. This paper is organized as follows. The section 2 presents the study context concerning the problems relating to hospital management. The section 3 introduces the proposed modeling methodology for performance evaluation by simulation. The section 4, present the used simulation framework (VLE). In section 5, we present an application concerning the modeling of a chronic renal dialysis unit using the modeling methodology (modeling and specification with UML). We conclude with some perspectives of research.

2 STUDY CONTEXT: PROBLEMS RELATING TO HOSPITAL SYSTEMS

Hospital managers have to control the health care consumption without altering their quality, so as to efficiently allocate human and financial resources to each unit in the best way.
For these systems from an experimental point of view it is nearly impossible to anticipate:

- the consequences of the new policy,
- the system reactivity faced to unexpected situations: important increase of the workload, breakdowns on equipment,…

Therefore hospitals are confronted with problems relating to:

- the dimensioning of the different health care units (type and number of resources),
- the understanding of the functioning of the system, as well as the control and the minimization of health costs,
- the efficiency improvement to increase the quality of services: a better allocation of resources, the length of waiting lists, the decrease in length of stays, etc.,
- the analysis of interactions between the units within the hospital organization,
- the characterization of the exploitation plan (new management and piloting policies),
- the activity and benefit planning (study of the usual and foreseeable workload) in order to anticipate human, equipment and financial resources and to meet patients' health care needs (for example the control and planning of admissions),
- the study of the system reactivity faced to particular conditions (riskiness on hospital facilities, staff decrease, important increase workload).

In the majority of the studies in hospital management, the various staff and equipments are considered as always available. This is not the case in reality. The equipments are unfortunately prone to failures and are inalienable during the maintenance actions; the staff might not be always available (due to planning limited by legislation for instance). Our approach could also take into account problems concerning breakdowns and maintenance of equipments, but this aspect is not described in this paper.

The depicted approach allows to use multi-modeling during the modeling phases. It is possible to use several diagrams of the UML methodology to generate one VLE simulation model. Each diagram allows describing corresponds to an aspect of the considered system like structural information or behaviors. It allows to describe the different levels with various formalisms.

The problems are strongly correlated. Thus, for example the study of a new management policy can be made with a view to improve the system efficiency and the service quality, and to emerge on to a new dimensioning of resources. The finality of such studies is based on the improvement of the service quality.

The need of an a priori or a posteriori evaluation of the system performances is essential. Providing to the managers with tools allowing them to reproduce the functioning of their systems seems an interesting idea. Indeed, the possibility of testing several configurations or different management and piloting rules would be useful due to the complexity of these systems and the encountered problems. Solving these problems requires to use adapted methods, techniques and tools. A study is performed to evaluate each system. It leads to the implementation of a model dedicated to the system, which cannot be used anymore (for another study or even for an analogous system). The presented methodology constitutes a proposal to overcome the problem relating to non-reusable dedicated models.
In this study we focus on the reorganization of a chronic renal dialysis unit.

3 MODELING TOOLS FOR HOSPITAL SIMULATION

3.1 Methodology

The meta-models have been designed in order to implement the reengineering software component base and to give a user guide of this base (model patterns). It is composed of three subsystem: **a *logical subsystem***: entities constituting the system workload and generating their activities; **a *physical subsystem*** owning all the physical entities (resources / care facilities) that handle activities generated by the system workload; **a *decision subsystem*** receiving information and converting it into activities (management and piloting rules). The explicit description of the meta-models does not fit within the scope of this article.
The proposed meta-models (one for each subsystem) are generic and have to be specialized to a system class, (for example hospital management class). Subsequently from the specialized model we have only to instantiate it in order to obtain the dedicated model (in our case, the cor-

responding model to dialysis unit). Modeling step lies on UML diagrams which are available in the majority of modeling softwares. The XMI file generated to store the model is used in MDA to generate generic and specific patterns. These patterns will be parameterized to obtain the simulation model.
Our methodology is divided into 3 steps (see Figure 1), according to the object-oriented paradigm.

1. *Object-oriented Analysis and Design*
This step is implemented for generic class of problems starting from the proposed meta-model. The aim is to create patterns with UML (allowing to automatically generate a reusable components base). A MMI skeleton (MMI: Man-Machine Interface) is designed for end users to elaborate their working model of the studied system.

2. *Specialization*
This step is implemented by a consultant who models the problems and determines if an existing model can be reused or customized.

3. *Instantiation*
This step is implemented by the end user, who models his configuration and tests several scenarios.

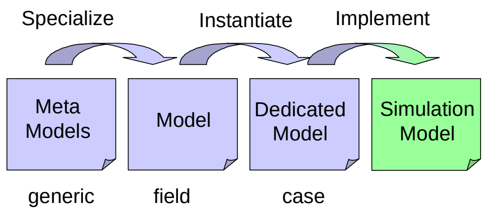


Figure 1. Generic methodology process

The target simulation language is VLE.

3.2 Methodological process

The proposed methodology is composed of six steps (see **Figure 2.** Details of the generic methodology). Step 1 concerns the analysis and the specification of the meta-models. The step 2 is performed for a generic system class relating to domain analysis; for example, the hospital management class (comprising systems with common characteristics). It consists in the specialization of the meta-models. The result of this domain analysis can be reused when a particular system of this system class is studied. In that case, only steps 3 to 6 are performed.

Step 1 *defines the classes of studied systems* (problems and objectives of the study) and concerns:

- the design of the meta-models for each subsystem (physical, logical and decision),
- the elaboration of the glossary and the user guide.

Step 2 *initiates the domain analysis; the aim is to design the specialized model from the meta-models*

We identify the characteristic entities relating to the studied system class. The relationships that the domain experts detect as fundamental elements are characterized. The domain analysis consists in identifying and taking into account all the common entities and in matching these entities into families (or classes) according to their common characteristics, functionalities and behaviors. Then we identify the links between the generic entities that are essential for the domain analysis.

We allocate entities to one of the three subsystems (logical, physical or decision subsystems) [Combes94, Combes01, Combes06]:

- ***logical subsystem***: a majority of flows that require the services of a hospital are relating to patients. These flows generate tasks (or activities) which are specific to each medical unit and correspond to the treatments undertaken by the physical subsystem entities;
- ***physical subsystem:*** confronted with the diversity of terms used in the different levels of the hierarchy, we put together the divided subsystems in terms of "functional and care units" and "administrative unit" respectively. From the perspective of administrative and functional units, we are able to take into account a descending or ascending hierarchical analysis of the physical subsystem. Each of these units has: staff resources (medical staff, administrative employees); material resources (functional elements, care units, conveying means, rooms/wards…) etc;
- ***decision subsystem***: it contains a set of rules concerning several logical and/or physical subsystem entities. The decision subsystem has the scope in:
  - modifying the physical subsystem evolution by its decisions,
  - determining the patient's flow path (route according to his needs and the system state),
  - applying the global and local rules of the physical subsystem resources.

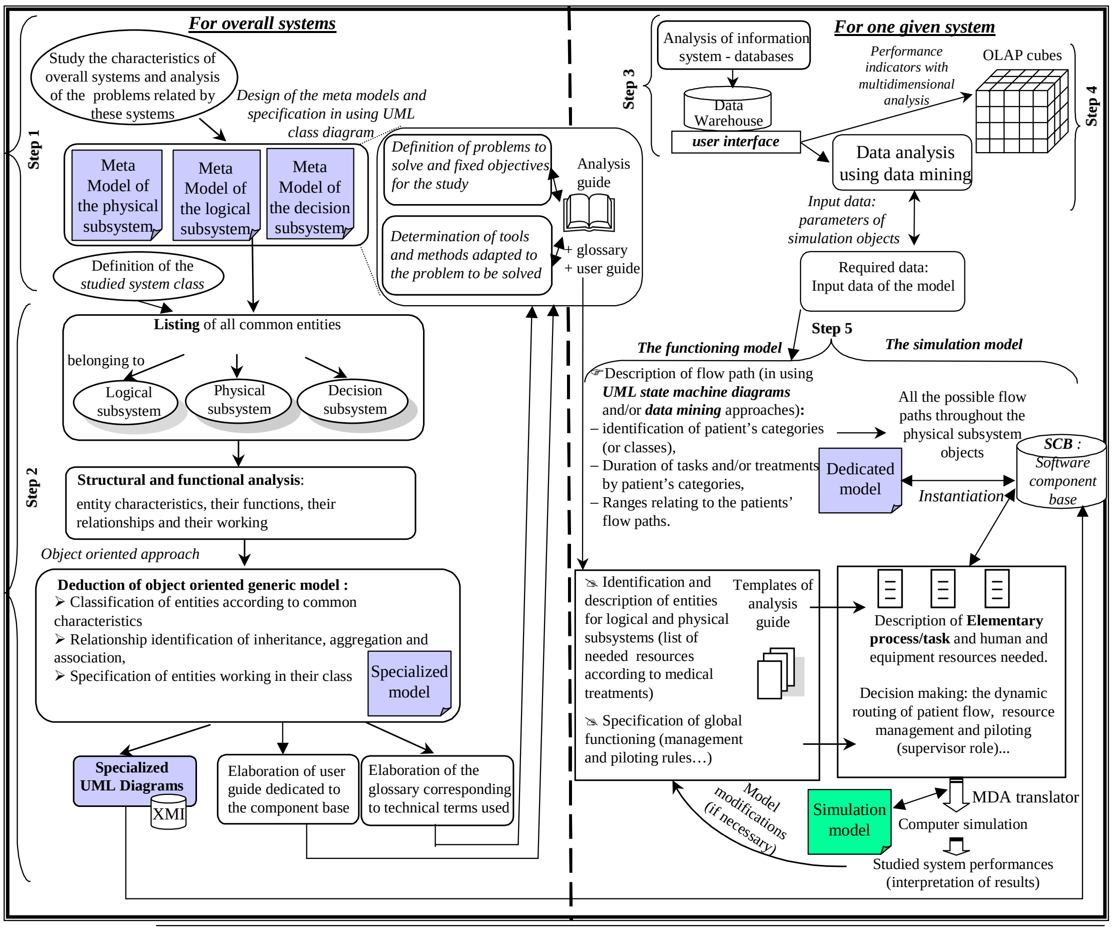

For overall systems
Step 1
Study the characteristics of overall systems and analysis of the problems related by these systems
Design of the meta models and specification in using UML class diagram
Meta Model of the physical subsystem
Meta Model of the logical subsystem
Meta Model of the decision subsystem
Definition of the studied system class
Definition of problems to solve and fixed objectives for the study
Determination of tools and methods adapted to the problem to be solved
Analysis guide
+ glossary
+ user guide
Step 2
Listing of all common entities
belonging to
Logical subsystem
Physical subsystem
Decision subsystem
Structural and functional analysis: entity characteristics, their functions, their relationships and their working
Object oriented approach
Deduction of object oriented generic model :
Classification of entities according to common characteristics
Relationship identification of inheritance, aggregation and association,
Specification of entities working in their class
Specialized model
Specialized UML Diagrams
XMI
Elaboration of user guide dedicated to the component base
Elaboration of the glossary corresponding to technical terms used
For one given system
Step 3
Analysis of information system - databases
Data Warehouse
user interface
Performance indicators with multidimensional analysis
OLAP cubes
Step 4
Data analysis using data mining
Input data: parameters of simulation objects
Required data: Input data of the model
Step 5
The functioning model
The simulation model
Description of flow path (in using UML state machine diagrams and/or data mining approaches):
– identification of patient's categories (or classes),
– Duration of tasks and/or treatments by patient's categories,
– Ranges relating to the patients' flow paths.
All the possible flow paths throughout the physical subsystem objects
Dedicated model
Instantiation
SCB : Software component base
Identification and description of entities for logical and physical subsystems (list of needed resources according to medical treatments)
Templates of analysis guide
Specification of global functioning (management and piloting rules…)
Description of Elementary process/task and human and equipment resources needed.
Decision making: the dynamic routing of patient flow, resource management and piloting (supervisor role)...
Model modifications (if necessary)
Simulation model
MDA translator
Computer simulation
Studied system performances (interpretation of results)

Figure 2. Details of the generic methodology

Step 3 *studies the existing information system*
The existing medical information system of the hospital is analyzed in order to design a data warehouse dedicated hospital supply chain and its user interface (if it is possible and it depends of available data).

Step 4 *specifies the patients' flow paths*
Two cases are taken into account. In the case where reliable databases are available and a data warehouse has been designed, data mining approaches can be used to identify the patients' profiles (or categories) and the associated flow path. The model of the patients' flow path can be deduced, but also, the estimation of the tasks duration and/or treatments by patient's categories. In fact, we do not have determinist range and each patient can be a particular case (undetermined range).
In the other case, we have to specify patients' flow path using UML diagrams (in particular the dynamic diagrams). In this article, we present this second case. In this step, we specialize the generic model concerning the logical subsystem.

Step 5 allows to *elaborate the dedicated model of a particular system from the results of the domain analysis*. In order to specify the functional and decisional models (working rules), we use the analysis guide, the glossary and the user guide of the software component base. This step consists in instantiating the specialized models. The resources required by a state or each activity are described. Therefore, the adapted patterns are parameterized to generate the simulation model. The different models need to be validated. In this step we also collect input data of simulation models (parameters and input data) and the performance indicators to be observed.

In ***Step 6***, the simulation model is generated and a validation phase is performed.

3.3 Using UML for specification (link between Step 2 and step 5)

The proposed modeling methodology uses UML formalism and is based on:
- the use case diagrams to describe the scenarios of each use case,
- the class diagrams (see figure 3) for the analysis and specification of entities constituting the system from the corresponding to the meta-model from which it is specialized and instantiated),
- the state machine diagrams to describe the internal behavior of each class and the sequences of activities.

Most UML modeling software can store diagrams using the XMI formalism. This explains why no specific software is required by this approach. The related XMI file is used to generate simulation patterns with the MDA.

Then the use of MDA approach allows to avoid the learning of a new language (simulation) and to reduce the adaptation phase of (generic) UML models to the specificity of the simulator. The translation from the UML diagram into the simulation language is mostly automatic. This approach facilitates the evolution of the model. Indeed we have just to modify UML diagrams in order to obtain the corresponding simulation model.

4 THE VLE FRAMEWORK

In this section, we present the VLE simulator and the underlying paradigm and formalism. This simulator relies on strong concepts and intrinsically provides multi-modeling capabilities. This perfectly matches the objective of the simulation and modeling tool that we are currently implementing. This is also largely facilitated by the available extensions such as UML statecharts.

4.1 VLE

VLE [Quesnel09] (Virtual Laboratory Environment) is a software and an API (Application Programming Interface) which supports multimodeling and simulation by implementing the DEVS abstract simulator.
VLE is oriented toward the integration of heterogeneous formalisms. Furthermore, VLE is able to integrate specific models developed in most popular programming languages into one single multimodel.
VLE implements the Dynamic Structure Discrete Event formalism (DSDE) (Barros97) which provides the abstract simulators for Parallel DEVS (PDEVS) (Zeigler00) for the parallelization of atomic models and Dynamic Structure DEVS (DSDEVS) (Barros96) for the M&S of systems where drastic changes of structures and behaviors can occurred over time. DSDE abstract simulator gives to VLE the ability to simulate distributed models and to load and/or delete atomic and coupled models at runtime. VLE proposes several simulators for particular formalisms. In addition to DSDE, for instance, cellular automata, ordinary differential equations, difference equation, UML statecharts and so on.
This framework can be use to model, simulate, analysis and visualize dynamics of complex systems. His main features are: multi-modeling abilities (coupling heterogeneous models), a general formal basis for modeling dynamic systems and an associated operational semantic, a modular and hierarchical representation of the structure of coupled models with associated coupling and coordination algorithms, coupling of pre-existing models, distributed simulations, a component based development for the acceptance of new visualization tools, storage formats and experimental frame design tools, and free and open source software.

4.2 STATECHARTS EXTENSION

A state diagram [Harel87, Harel98] is a type of finite state automaton used in system modeling and in computer science. It describes the dynamical behavior of systems. In UML language, this kind of diagram is known as UML statechart [OMG09] (or UML state machine). It is an object-based variant of Harel statechart.
Modeling systems based on discrete and finite states is to represent the system as discrete states and transitions linking states. Transitions represent changes of state on triggering events and may support actions. An action is a function applied on the system state other than discrete states. In the case of UML statecharts, actions represent the methods of objects. UML statecharts have the characteristics of both Mealy machines [Mealy55] and Moore machines [Moore56]. They support actions that depend on both the state of the system and the triggering event, as in Mealy machines, as well as entry or exit actions (see Figure 3), which are associated with states rather than transitions, as in Moore machines.
UML statecharts introduce the new concepts of hierarchically nested states and orthogonal regions, while extending the notion of actions and events. A state (called superstate) can be decomposed to several states (substates). The events are handled by substates or superstate if the substate does not prescribe how to handle the event. UML statecharts also introduce the AND-decomposition: a composite state can contain two or more orthogonal regions and each region can be active simultaneously.

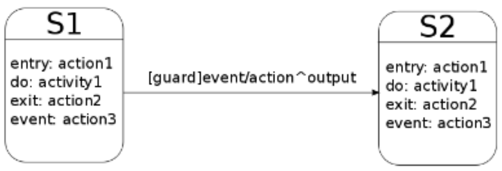


Figure 3: UML statechart with two states and a transition representing the different available concepts in VLE platform

Every state can have optional entry actions (e. g. action1), which are executed upon entry to a state and optional exit actions (e. g. action2), which are executed upon exit from a state. When an event occurs, a transition can handle it or a state (e. g. action3). In this last case, an action is executed without state change. The concept of activity is introduced to define a continuous action in state. While the state is active, the activity is executed.
A transition can also be triggered on a condition: the guards. Guard conditions are boolean expressions evaluated dynamically based on the value of extended state variables and event parameters. Moreover, when an event occurs, the state machine responds by performing actions and an action can be a generation of another event instance.
In VLE, all concepts are defined except for hierarchically nested states and orthogonal regions.

5 AN APPLICATION: THE MODELING OF A CHRONIC RENAL DIALYSIS UNIT

The aim of this study was to show one application of our modeling and simulation tool, in the particular case of a chronic renal dialysis unit. The description of the working system is due to the article of R.M. Korabick [Korabick78].
With the methodological framework we have modeled a medical unit specialized in renal dialysis treatments. This pathology is severe. The patients' scheduling is a critical component of management because the chronic character of the pathology obliges them to come several times a week to have a treatment. The normal scheduling of the treatment is:
1) The patient is installed and connected to the machine by a nurse (25 minutes),
2) The treatment lasts 8 periods of 30 minutes. Between each period, the nurse comes to monitor the good execution of the treatment (10 minutes without corrective action),
3) The patient is disconnected from the machine by the nurse (25 minutes).
In addition, before each treatment, the nurse has to prepare the machine (30 minutes). After each treatment, she has to clean the machine (30 minutes).
To insure the good quality of the treatments, corrective actions are done on each dialysis machine every 4, 8 and 20 treatments. A corrective action requires one or two nurses; it is done during the treatment monitoring.
The labor cost is typically the most substantial cost for the operating unit. R.M. Korabik [Korabick78] has evaluated this part to 30% to 40% of the total expenses. So it would be interesting to control labor cost and to increase equipment utilization through good scheduling of employees and patients.

5.1 Expression of hospital needs for the chronic renal dialysis

The first step consists in the capture of functional needs. The UML formalism taken into account is:
- Packages in order to identify the different models (for simulation, scheduling…),
- Use cases which describe for each package, the description of system behavior of user point of view,
- Diagram of classes for each use case,
- Diagram of sequences which document each use case (scenarios of each use case).

Identification of packages

We identify 4 packages (see figure 2):
- "*Patient Management*" modeling management of patients' medical records and patients' appointment,
- "*Dialysis treatment*" describing the working of a chronic renal dialysis,
- "*Logistic*" taking into account the management of equipment resources,

"*Staff management*" concerning the management of staff timetable and work notice board for each staff categories.
The corresponding use cases of each package are given in figure 3.

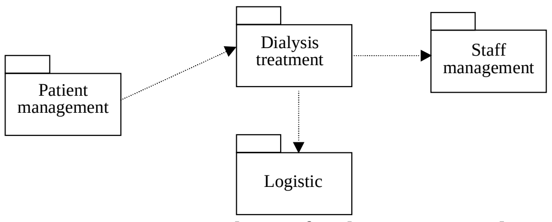
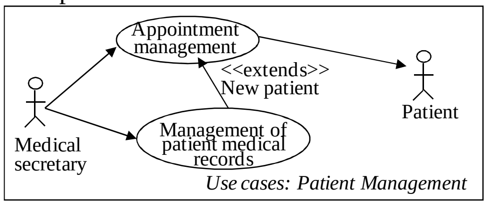
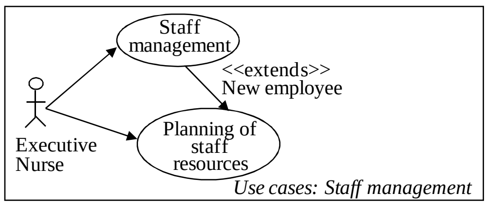
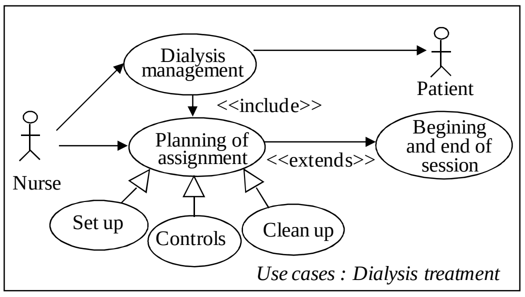
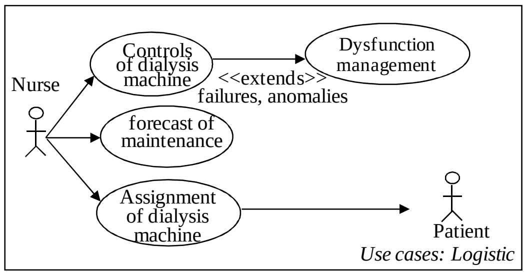


Figure 4: UML Packages for the current study.








Nurse
Controls of dialysis machine
Dysfunction management
<<extends>> failures, anomalies
forecast of maintenance
Assignment of dialysis machine
Patient
Use cases: Logistic
Figure 5: The use cases

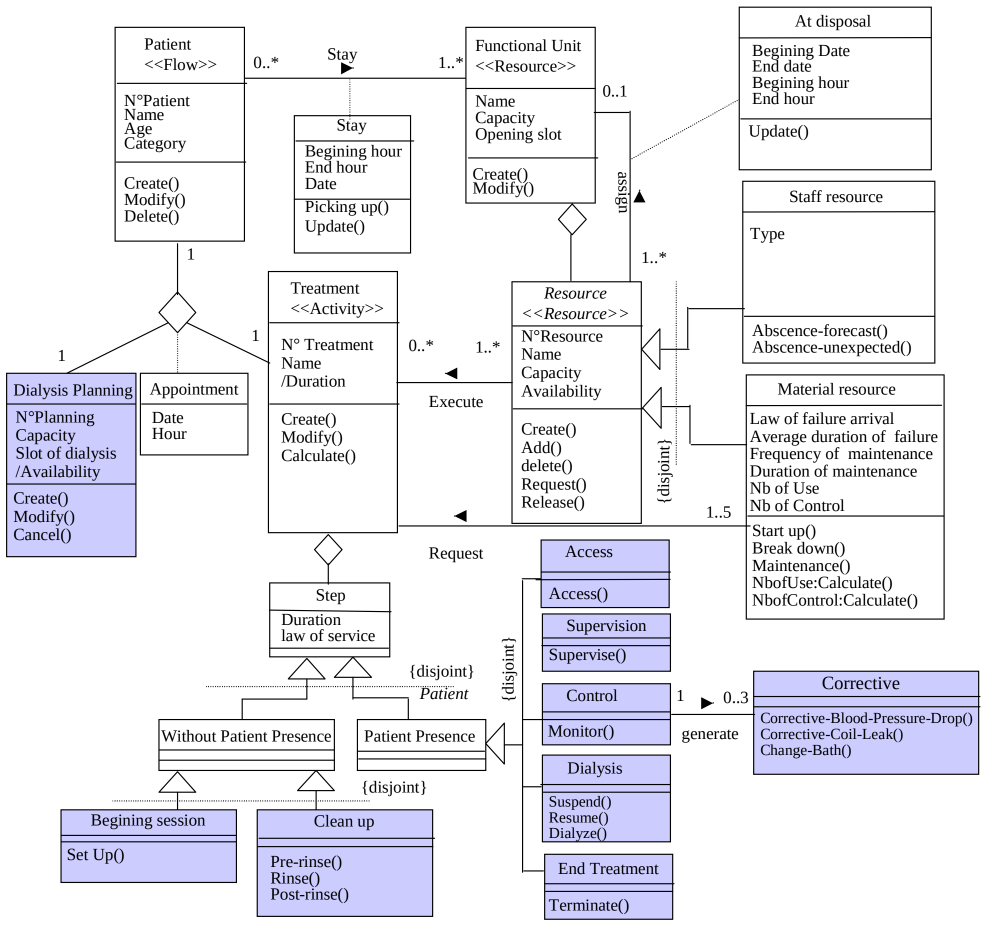


Figure 6: The UML classes diagram

Now, we present (figure 4) the classes diagram corresponding at the use cases "*dialysis treatment*".

5.2 Specification of the dynamic model

The aim is to formalize the scenarios involving the objects exchanging messages and the internal behavior of the classes.
Figure 5 presents a scenario of the use case "*planning assignment*"

Figures 6 and 7 describe the internal behavior of the class called "*Treatment*".

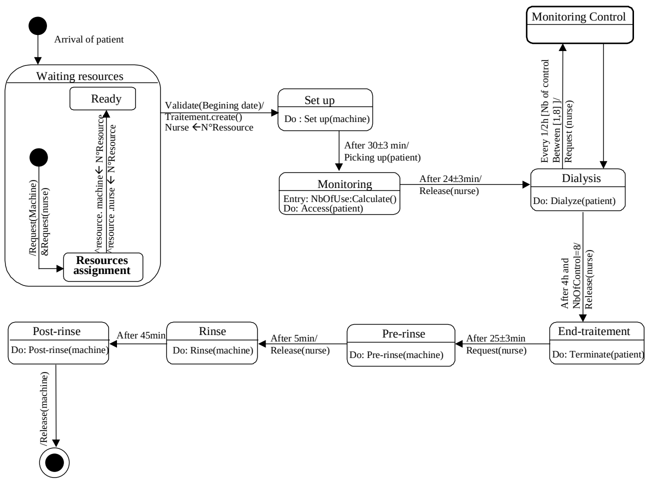


Figure 7: Dynamic diagram of "treatment" class

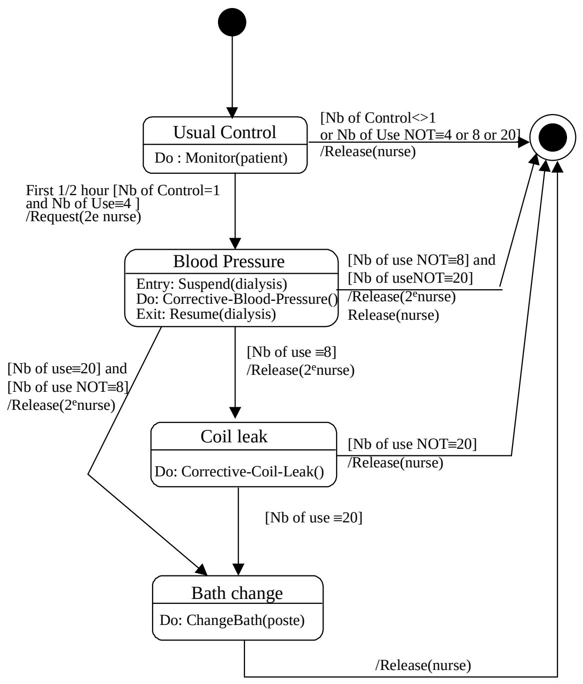


Figure 8 : Dynamic diagram of "Monitoring control" macro-state

5.3 Simulation

The models can be automatically translated into the simulation language. In this case, only the most specialized classes are taken into account to generate simulation model. Statechart diagrams are used to create simulation patterns to mimic the dynamic of the studied system.
The modeling tool implements a translator package, which is used to generate the simulation model from the UML diagrams (or XMI file). VLE facilitate this translation of UML diagram with its statechart extension.
Indeed, thanks to its underlying multi-model paradigm, VLE permits to easily translate each package into a model that can communicate with other models thanks to input and output ports. Each of these models might in turn be a composed-model or an atomic-model. Each atomic-model might be expressed either directly in DEVS or through one of the available extensions of VLE (statecharts in the presented example). Moreover the structured C++ syntax of VLE allows to easily modify the simulation model.
The XML file needed for the parameterization is generated with the VLE simulation model. It allows to customize/tune the model to evaluate several scenarios for instance.
In this particular case, dealing with the reorganization of a chronic renal dialysis unit, the performance indicators are waiting times, idle time, number of patients, workload of nurse, schedule appointment...
A graphical interface could be implemented for the end user in order to value the simulation model according to the studied case, like number of dialysis machines, number of employees, reliability of equipments.

6 CONCLUSION

We have presented an application of our modeling tool based on UML model to generate simulation models in the case of a chronic renal dialysis unit.
With the use of UML, the proposed modeling tool is not limited to the hospital domain and can be applied to the industrial field.
All components, needed for the modeling methodology, are implemented except for the decisional subsystem. The latter is in progress for validation. The decision subsystem needs to improve integration of performance measure for the system.

7 REFERENCES

Barros, F. J., 1996, Dynamic structure discret event system specification: Formalism, abstract simulators and applications. Winter Simulation, 13(1):35-46.

Barros, F. J., 1997. Modeling formalisms for dynamic structure systems. ACM Transactions on Modeling and Computer Simulation (TOMACS), 7:501-515, october.

Combes C., 1994, "Un environnement de modélisation pour les systèmes hospitaliers", *Ph.D. Thesis*, University of Blaise Pascal, Clermont-Ferrand II, France.

Combes C., 2000, "Contribution of modeling and simulation to health care systems", *Heath and System Sciences*, Edition Hermès Science, N°4, pp 1-31.

Combes C. N. Meskens, H. Fei, M. Gourgand, 2006, "A methodological framework dedicated to hospital supply chain management with the combination of data warehousing and data mining", ILS'06, *International Conference on Information Systems, Logistics and Supply Chain*, Lyon France, May 15-17.

Harel, D., 1987, Statecharts: A visual formalism for complex systems. Science of Computer Programming, 8(3):231–274, June.

Harel, D. and M. Politi, 1998, Modeling Reactive Systems with Statecharts, the STATEMATE Approach. McGraw-Hill. p. 258.

Korabik R. M., 1978, "A New Approach to Operational Efficiency for Chronic Renal Dialysis", Winter Simulation Conference I.E.E.E, Highland, pp 659-662.

Law A.M., W.D. Kelton, 1999, "Simulation Modeling and Analysis", Edition Mac Graw Hill Book Company.

Mealy, G., 1955, A Method to Synthesizing Sequential Circuits. Bell Systems Technical Journal. pp. 1045–1079.

Moore, E.F., 1956, Gedanken-experiments on Sequential Machines. Automata Studies, Annals of Mathematical Studies, 34, 129–153. Princeton University Press, Princeton, N.J.

Mukerji J.and J. Mille, 2001, MDA Guide Version 1.0.1, Overview and guide to OMG's architecture, http://www.omg.org/cgi-bin/doc?omg/03-06-01, 2001.

OMG (Object Management Group), 2004, Unified Modeling Language Specification, Version 1.4.2, http://www.omg.org/cgi-bin/doc?formal/04-07-02.

OMG (Object Management Group), 2009, OMG Unified Modeling Language (OMG UML), Superstructure Version 2.2, http://www.omg.org/spec/UML/2.2/Superstructure/PDF, February.

Quesnel, G., R. Duboz and É. Ramat, 2009, The Virtual Laboratory Environment - An Operational Framework for Multi-Modelling, Simulation and Analysis of Complex Systems, Simulation Modelling Practice and Theory, (17), 641-653, April.

Zeigler, B. P., D. Kim, and H. Praehofer, 2000. Theory of modeling and simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems. Academic Press.