Preprints of the 18th IFAC World Congress
Milano (Italy) August 28 - September 2, 2011

Improving the software development life cycle in process control using UML/SysML

Idilia Batchkova, Iskra Antonova

Department of Industrial Automation, University of Chemical Technology and Metallurgy, Sofia, Bulgaria (e-mail:{ idilia, iskra.antonova}@ uctm.edu)

Abstract: The transition from approaches based on a directly code creation to model-driven software development poses the modeling as one of the first most important things in the all field of engineering. The main aim of the presented paper is to use the current advantages in the UML extensions and profiles in order to improve the software development life cycle in the field of process control. The proposed approach is based on the combined use of UML profile for system engineering SysML, IEC-61499 standard for development of distributed control systems and modified Harmony SE methodology. The suggested approach is illustrated with a simple example for development of tank level feedback control system. Finally some conclusions are made.

Keywords: process control, system engineering, modeling, software process model, MDE, UML, SysML.

1. INTRODUCTION

Since the emergence of first computer-based automation and control systems in the early 60's of last century, the development of automation and control constantly follows the progress in information and communication technologies. One of the trends in this direction is associated with the significant growing of the software used in control systems, its volume and complexity, the need to integrate this software into physical systems and to ensure its integrity, security, reliability, interoperability and portability. The response to these new challenges is the so-called Model Driven Engineering (MDE) (Kent, 2002) or its counterpart in the field of architectures - Model Driven Architecture (MDA) (OMG-MDA, 2003). The approaches of MDE and MDA consist in transformation of different platform independent models towards executable applications. The transition from approaches based on a directly code creation to model-driven software development poses the modeling as one of the first most important things in the all field of engineering. The Unified Modelling Language (UML) (OMG-UML, 2010) as a general purpose modelling language and an open standard supports the MDE and MDA. It does not specify a methodology for software or system design but aims to provide an integrated modelling framework, covering structural, functional and behaviour descriptions. The UML notations based on different diagrams focuses multiple aspects of a complex system and complexity need for detailed analysis and design.

In the last years there has been an increasing striving to use UML in control, automation, and industrial enterprise engineering. There are many working groups whose research activities are directed in filling the gap between state of the art in software engineering and state of the practice in the control application domain. They use different approaches and techniques in order to extend the object-oriented software development and especially UML to development of real time systems. A short overview of the existing real-time extensions is presented in part 2 of the paper. Main drawbacks of all these extensions are connected with the difficulties in modelling and analysis of the closed loop systems. In order to overcome these drawbacks, in this work, the UML profile for System Engineering (SysML) is used (OMG-SysML, 2006).

SysML is built as a response to the OMG RFP (Object Management Group Request for Proposal) and intends to support the modelling of a broad range of systems, which may include hardware, software, data, personnel, procedures, and facilities. It is a "standard modelling language for systems engineering to analyse, specify, design, and verify complex systems, intended to enhance systems quality, improve the ability to exchange systems engineering information amongst tools, and help to bridge the semantic gap between systems, software, and other engineering disciplines" (OMG-SysML, 2006).

The main aim of the proposed study is to improve the software development life cycle in the field of process control through MDE approach, combining the use of modified Harmony methodology (Estefan, 2008), IEC-61499 based concepts, reference architecture and models (IEC61499, 2005), and the UML profile for system engineering SysML. Such a way the development process will allow an early defining of requirements to the control system, enabling the verification and validation processes through modelling the closed loop control system and achieving of modular, re-usable, open and vendor independent control applications, characterized through the three main features: interoperability, portability and configurability.

The paper is organized in 4 parts. After the introduction, in part 2 the real time extensions of UML are analysed and discussed. In part 3 of the paper, the MDE approach for improvement of the software development live cycle model in process control is presented. Part 4 presents a simple case study concerning an application of the suggested approach for water level feedback control. Finally some conclusions are drawn and some aspects of the future research are discussed.

2. STATE OF THE ART IN USING UML IN THE FIELD OF CONTROL

The main challenges towards real-time UML are connected with the creation of different mechanisms to handle real-time features such as: models of physical time, timing specifications, timing facilities, modelling and management of physical resources and concurrency. Another important issue is the development or use of means for early verification and validation of the designed systems in respect not only to their functionality, but also in respect to the non-functionality requirements. Further in this part we try to summarize and analyze the research activities and current state of the art in these fields in order to select and apply the most suitable and effective results of them in the software development life cycle model for the field of process control.

2.1. Real Time extensions to UML

There are many different proposals for extending UML to support the design and analysis of real-time systems and they can be joined in 2 main abstract classes, shown in Fig.1. The first class of real-time (RT) extensions is connected with the combination of standard UML capabilities with those of other real-time frameworks, languages and methods, covering different real time features of the designed systems. One very large subclass of RT extensions of UML is connected with filling the gap from the lack of well-defined formal semantic and ensuring of capabilities for formal verification of designed UML models. In general, these extensions can be separated into two major categories. The first of them is based on defining of the semantic domain and giving a sound semantics to the graphical notations in this domain. Examples of such profiles are TURTLE (Apvrille *et al.*, 2004) and rtUML (krtUML) (Damm *et al.*, 2002). The plentiful graphical notations however make the semantic domain very complex, and impede the analysis tool support. This obstacle gives an advantage to the more workable approach including a combination of graphical notations and formal specification languages, and exploring the formal reasoning capabilities of formal methods. There is variety of works in the field of software engineering based on well-established traditional formal methods such as VDM, Z, B, and their object-oriented extensions such as VDM++, Z++, Object-Z etc. (Mwaluseke and Bowen, 2001). One of the mostly used combinations in the development of RT applications is that between UML and SDL, which share a number of qualities, like having a graphical notation, good readability and good tool support. They also incorporate object orientation and state machines, which make UML and SDL suitable to work together [Verschaeve, 2001].

Typical representative of the approaches combining UML with other methodologies is the UML-RT profile of IBM Rational Rose (Selic and Rumbaugh, 1998). It is based on some concepts of the ObjecTime ROOM methodology and provides three principal constructs (capsules, ports, and connectors) for modelling the structures of a real-time system using the basic UML 1.4 mechanisms of stereotypes and tagged values. The main shortcomings of UML-RT are in specification of time constrains and timeliness properties because there are not means for it. It is more suitable for modelling the structure and communications between the different elements in the system. Another very promising approach especially applied for development of distributed real-time systems in the control and automation domain is the collaborative use of UML and the reference architecture and models proposed in the new IEC-61499 standard (IEC-61499, 2005). A short overview of the proposals in this field are presented and analyzed in part 2.3 of the paper.

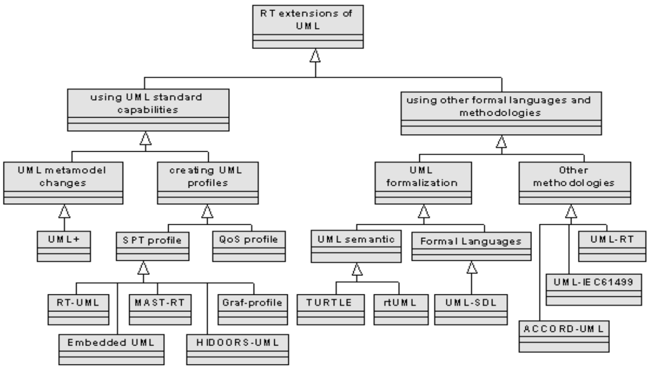

Fig.1. UML extensions for real time

The second class of extensions follows the idea of Douglas (1998) and Selic (2003) that the behaviour of complex real-time control systems can be fully described by using the standard capabilities of UML and is subdivided in two classes applying the following extension strategies:

- Creating UML profiles (standard and specific) on the base of stereotypes, constraints and tagged values. The 3 built-in extension mechanisms can be used separately or together.
- Changing the UML meta-model by explicitly adding new meta-classes and other meta-constructs.

The objective of UML profiles is to package specific terminology and substructures for a particular application domain. One of the first attempt to provide RT capabilities of UML in this direction is the OMG initiative for creating of the profile for Schedulability, Performance, and Time Specification (SPT-profile) that is proposing a framework to model quality of service, resource, time and concurrency concepts in order to support predictive quantitative analysis of the UML models (OMG-SPT, 2003). This profile supports two well-established forms of time-based model analysis: schedulability analysis based on schedulability theory and performance analysis based on queuing theory or Stochastic Petri Nets. The SPT profile is used as a basis for some other UML profiles some of them are RT-UML (DiPippo, 2000), MAST-RT (Medina *et al.*, 2001), embedded UML (Martin *et al.*, 2001), HIDOORS UML profile (Meinier *et al.*, 2004), Graf-Ober profile (Graf and Ober, 2003) etc. The new UML MARTE profile, supporting the specification, design and verification/validation in the model driven development of real time and embedded systems, will replace the SPT profile with the appearance of the new UML2.3 version (OMG-MARTE, 2006). The UML profile for modelling Quality of Service and fault tolerance characteristics and mechanisms (QoS-profile) adopted by OMG in 2004 supports the implementation of SPT-profile through the definition of a catalogue with different categories of QoS characteristics, Quality models and UML model annotations using QoS requirements (OMG-QoS, 2004).

Extensions to the second subclass are a priority of OMG, but there are other initiatives too as for example the UML+, where new elements are added to the UML meta-model in order to achieve RT capabilities (Bianco *et al.*, 2007). The current work on the evolution of UML standard at OMG aims the integration of the most successfully contributions to the real time issues. UML2.x provides means for architectural modelling inspired from UML-RT (Selic and Rumbaugh, 1998), with structured classes, ports (isolate an object internals from its environment), connectors (which link communicating ports) and protocols (defined, reusable, interaction sequences). It presents some features that support real-time aspects and they may be summarized as follows:

- Modelling of concurrency through introduction of active objects, concurrent composite states and concurrent operations.
- Expressing of timing behaviour of the system through introducing of new diagram (Timing Diagram) and new data types (*Time* and *TimeExpression*).
- More open to formalization, including more run time semantics, better semantic definition of the state and activity diagrams.
- Better definition of components and subsystems through new graphical features for the class diagram and new notations for the existing diagrams.
- Improved extension mechanism in respect to the user-defined meta-classes and relations, and profile mechanisms.

The main conclusions to be imposed by the considered and analyzed approaches for real time extensions of UML may be summarized as follows:

- There is a strong interest worldwide on the real-time extensions of UML and their use in different applications domain;
- Current proposals for real-time UML are insufficient and not completely compatible. Most of the proposals provide a good support for parallelism modelling but are poor to express quantitative real-time features (such as deadlines, periods and priorities). There are many proposals based on proprietary solutions, which are not fully compliant with the UML standard.
- Despite the variety of structural and behaviour improvements, UML2x has insufficiently support for development of process control system in respect to expressing hardware-software interdependences and timing models. That is why the using of specialized profiles based on UML2.x is the best solution for these purposes.

2.2. SysML

SysML is a general-purpose modelling language for system engineering that reuses a subset of the last UML2.x versions and provides additional extensions through stereotypes, diagram extensions and model library in order to model a wide range of system engineering problems as for example specifying requirements, structure, behaviour, allocations and constraints on system properties to support engineering analysis. The reusable subset of UML, known as UML4SysML includes Interactions, State machines, Use Cases and Profiles. In Fig.2 the set of SysML diagrams in respect to their modelling aspects is summarized. The system structure design is supported by four types of diagrams: Block Definition Diagram (BDD), Internal Block Diagram (IBD) reinforced by Parametric Diagram (ParD), and Packages Diagrams (PacD). The Behaviour Diagrams incorporate four diagrams too, namely: Activity Diagram (AD), Sequence Diagram (SD), State Machine Diagram (SMD), and Use Case Diagram (UCD). The Requirements Diagrams (RD), which can be presented in graphical, tabular or tree structure format, are used to specify different constructs for system requirements and to cover the relationships between them. In SysML two kinds of requirements are used – functional and performance, as they specify the capabilities or the conditions which must be performed or satisfied by the system.

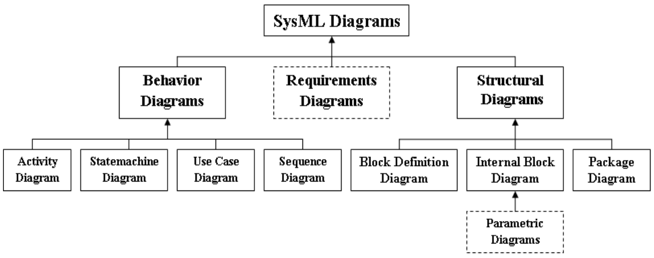

Fig.2. SysML Diagrams (OMG-SysML, 2006)

Other modelling capabilities of SysML, not shown in fig.2 are the cross-cutting constructs, such as allocations for connecting of different views, and Profiles & Model libraries allowing further customizing and extending SysML to specific applications. SysML also includes extensions supporting the causal analysis, the verification and testing processes and the decision tree development.

The advantages of SysML compared with those of UML2.x to the modeling of control systems can be summarized as follows:

- SysML supports the whole development life cycle of control engineering applications from the requirements definition to the software implementation;
- Based on extended activity diagrams, parametric diagrams and flow ports and items, the proposed approach may be applied for continuous and hybrid systems;
- The possibilities to model the physical systems in detail enhance the procedures for analysis, testing and validation of designed closed loop behaviour of the system.

2.3. Combined use of UML and IEC-61499 standard

IEC-61499 standard defines the basic concepts, reference architecture and models for development of modular, re-useable and open distributed process measurement and control applications (Lewis, 2001). The main building block in an application according this standard is the function block (FB) and the control system may be modelled as logically connected function blocks through their input and output data and events. The proposed standard is purely functional and does not exploit the benefit of Object Technology. In order to extend the standardized concept to the whole development life cycle of control application and to use the all benefits of object-oriented paradigm some researchers undertake different extensions in UML notations according the concepts of IEC-61499 standard (Thramboulidis, 2004). These notations may be used as a modelling language to visualize, specify, construct and document the different elements of IEC-61499 based control system. Some researchers offer the control system to be designed in UML environment and then the models can be transformed based on well-defined rules in Function block based models using FBDK. A new model transformation language UML-FB is defined (Dubinin and Vyatkin, 2004). The both methodologies combine UML and IEC-61499 standard using the old UML1.4 version and the IBM Rational Rose tool.

3. SHORT DESCRIPTION OF SUGGESTED APPROACH

The suggested software development life cycle model uses combination of modified Harmony-SE methodology, which is a subset of the large methodology for integrated system and software development process (Estefan, 2008) and the concepts, reference architecture and models of IEC-61499 standard. The task flow development process includes the following basic elements as shown in Fig.3: Requirements Analysis; System functional analysis; Architecture design; Hardware/Software design specification. The proposed model-driven approach is based on SysML structure diagrams using blocks as basic structural elements. The requirements analysis phase starts with the translation of customer requirements into a set of requirements defining what the system must do and how well it must perform. For the purposes of this first stage the SysML requirements diagrams (RD) creating taxonomy of the captured requirements and System Use Cases (SUC) are used. The system functional analysis phase is presented through transformation on the identified functional requirements into coherent system functions. Use Case Diagram (UCD) presents the system functional phase. For each use case the analysis is performed. BDD and IBD are applied to present the composite system structure. Different Black-Box Activity (BB-AD) and Sequence diagrams (BB-SD) are used to capture the high level system functionality. The Architecture design is the third stage in the proposed methodology and includes the design of subsystem structures and behaviour based on White-Box (WB) variants of the diagrams used in the previous stage (WB-UCD, WB-AD, WB-SD). Other important tasks are the ports and interfaces definitions and the description of subsystem state-based behaviour using the SysML statecharts (SCD). In order to model the control subsystems at the architectural design level the concepts, reference architectures and models of IEC-61499 standard are followed. There is a clear correspondence between SysML and IEC-61499 standard. The modelling concepts in both standards share many similarities. For example FB and parts are analogous, IEC-61499 data and event interfaces and ports are similar too. The Execution Control Charts are replaced by statecharts. The different algorithms may be modelled through activity diagrams and allocate to different states of the statecharts. The last stage is the Hardware/Software design specification and is closely connected with the system implementation. All methodology stages include verification and validation tasks, based on the model developed in the previous stages and the defined requirements.

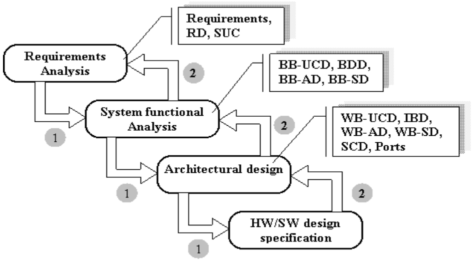
Requirements, RD, SUC
Requirements Analysis
2
BB-UCD, BDD, BB-AD, BB-SD
System functional Analysis
1
2
WB-UCD, IBD, WB-AD, WB-SD, SCD, Ports
Architectural design
1
2
HW/SW design specification
1

Fig.3. Illustration of the proposed approach

4. CASE STUDY

The application of suggested approach will be illustrated with a simple example of water level control in a tank as shown in Fig.4-1. This application demonstrates the modelling of control system from two different viewpoints – structural and behaviour. The discussed example is divided in two parts – physical system named "TankSubSystem"(Fig.4-3) and control system named "ControlSubSystem" (Fig.4-2). The control system is a feedback controller, which controls the level in the tank according the PID principle.

4.1. Modeling the physical subsystem

One of the big advantages of SysML profile is the possibilities to model physical systems. The physical system consists of a cylindrical tank, filled with water until specified level that is registered with a level sensor and is controlled by changing the input valve position. The static composite structure of the "TankSubSystem" is presented with BDD and shows the physical structure in terms of installed equipment.

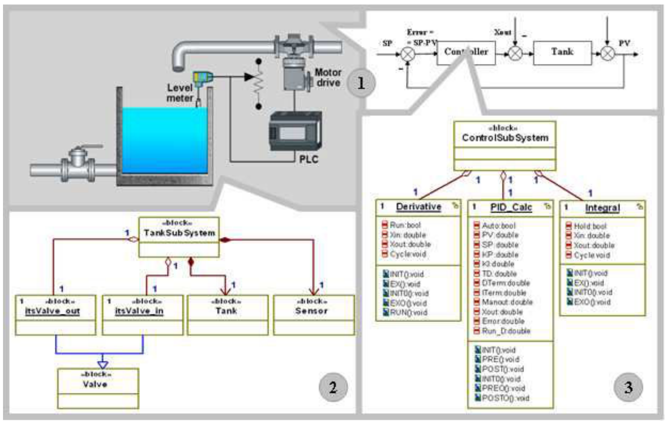

Fig.4. BDD of the subsystems

In Fig.5 the IBD of "TankSubSystem" is presented, which is composed by different types of equipments such as input valve, output valve, tank and sensor. Each part is defined uniquely through its attributes and operations. The attributes characterizing "Valve_in" and "Valve_out" are respectively input/output pressure (Pin/Pout), input/output flowrate (Qin/Qout), Cin/Cout for density of liquid and Sin/Sout for coefficient of input/output valve. The connections between parts are modelled as flow ports. In order to present the dynamical behaviour of the plant, the elements of "TankSubSystem" are described with their transfer functions as operations using Parametric diagram.

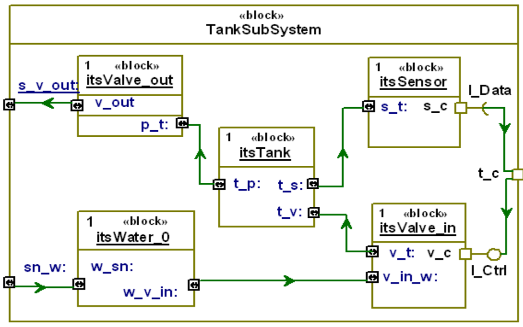

Fig.5. Static structure of the "TankSubSystem"

Parametric constraints are added to the model in order to support the analysis tasks and to describe the physical systems with equations. In Fig.6 the parametric constraints for the "TanksSubSystem" are presented in Parametric Diagram through the transfer functions for tank, input and output valves. The transfer function describing the tank includes the following parameters – T, Kin and Kout which are presented as attributes. These parameters may be calculated using indicated equations.

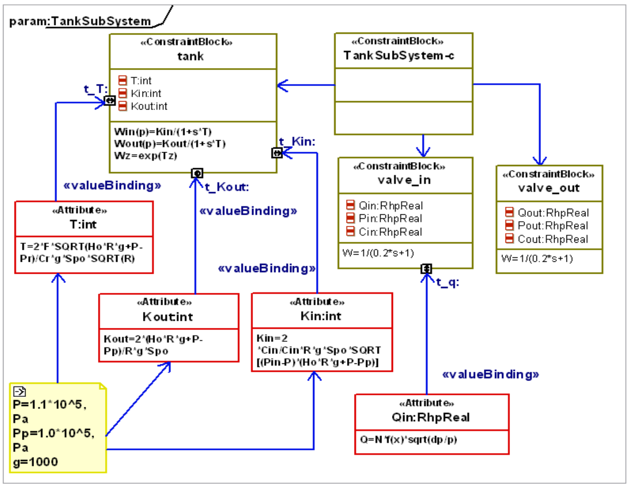

Fig.6. Parametric Diagram

4.2. Modelling the control subsystem

The "ControlSubSystem" is modelled according to IEC-61499 standard using 5 different models – system model, device model, resource model, application model and function block model. It is assumed that a simple PID algorithm is used to control the level in the tank. It has a number of parameters that must be tuned to match the process under control. The Internal Block Diagram named "ControlSubSystem" contains three main parts: "PID_Calc", "Derivative" and "Integral", which are interconnected via ports as shown in Fig.7-1. "Derivative" and "Integral" parts provide respectively derivative and integration function for inputs of real data. The "PID_Calc" part encapsulates the PID algorithm and calculates new output values which are proportional to a weighted summation of the error, the derivative error and the integral of the error. The parts are uniquely presented through attributes and operations. Their system behaviour is specified with statecharts. During the execution of the PID algorithm, "Derivative", "PID_Calc" and "Integral" parts change their execution states. The architectural design of "PID_Calc" part is illustrated in Fig.7. It has two execution states namely PRE and POST (Fig.7-2). The first state runs an algorithm to calculate the error between the set point and the process value. When an output event PREO is generated then the trigger execution of the derivative and integral blocks are executed. The POST state is triggered by input event POST and is entered when the external derivative and integral blocks have completed their execution. In UML/SysML the executable algorithms included in states are modelled by activity diagrams. The algorithm of PREO for the "PID_Calc" part is shown in Fig.7-3.

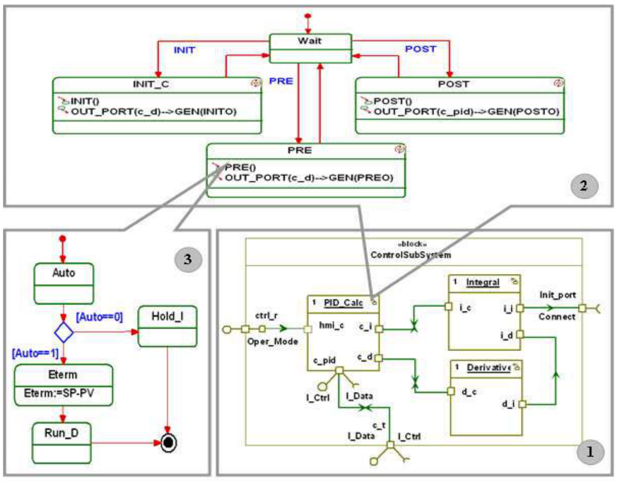

Fig.7. Modelling the "ControlSubSystem"

4.3. Modelling the closed loop control system

The first stage in the design of the closed loop control systems according the proposed approach is the definition of system requirements. Requirements analysis starts with analysis of the process inputs. In SysML profile all requirements are described textually and may be presented graphically. A requirement represents the behaviour, structure, and/or properties that a system, component, or other model element must satisfy. The functional requirements assigned to the control system are presented in Requirements Diagram shown in Fig.8-1 and they are defined as follow: the level set-point is 200 mm, the sampling period is 0.01 seconds and the output flow rate is not controlled. The specific operational aspect of the control system and the interactions between control system and its users are presented through Use Case Diagram, shown in Fig.8-2. The discussed control system may operate in different operation modes such as automatic mode, manual mode, reset, filling, emptying and sense which require different sequences of control actions to be executed and/or initialized. The static composite structure of the closed loop control system consisting of HMI, “ControlSubSystem”, “TankSubSystem” and communication bus, and their connections are presented with BDD (Fig.8-3). The message exchanges among the system blocks are presented by WB Sequence Diagram (fig.8-4). After sensing of the current level in the Tank, the “ControlSubSystem” read these data, calculate the movement of the valve and manages the actuator (Valve_in). The messages representing the “Valve_in” control are “Move_Pos1”, “Hold” and “Move_Pos2”. They depend on the tank level. In UML/SysML the communication between system’s parts is realized by different type of ports, which are special kind of parts giving access to the internal structures from the outside of a composite object.

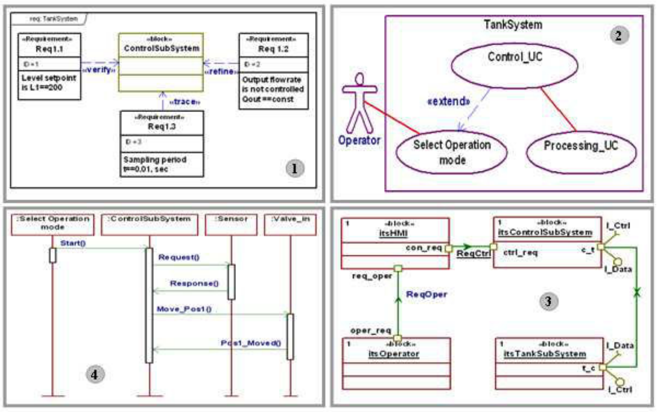

Fig.8. Development of closed loop control system

CONCLUSIONS

The main benefits of using the proposed approach can be summarized as follow: it provides comprehensive consistency in the syntax and underlying semantics; increases the potential and likelihood of reuse; supports the whole software development life cycle in the field of process control - from the requirements definition to the software implementation. Including the main features of SysML, based on extended activity diagrams, parametric diagrams, flow ports and items to the proposed approach opens the possibilities for modelling of continuous system and support the development in field of process control.

The future research activities are concentrated mainly in two directions. The first one aims to extend the approach to integrate different formal techniques for verification and validation of the developed models at different level of abstraction through meta-model based transformations. The second direction is connected to the last development level of the approach – HW/SW design specification. In order to better support this life cycle stage the MARTE profile of UML is planned to be integrated in to the proposed model for process control system development.

Acknowledgment: This work has been partially supported by the Bulgarian National Research Fund to the Ministry of Education and Science in the frame of Project BУ-307/2006.

REFERENCES

Apvrille, L., Courtiat, J.-P., Lohr, Ch. and Saqui-Sannes, P. (2004). TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit, *IEEE Transactions on Software Engineering*, Vol. 30, No. 7, July 2004, pp.473-487.

Bianco, V., Lavazza, L., Mauri, M. and Occorso, G. (2007). Towards UML-based formal specifications of component-based real-time software, *Int. Journal on Software Tools for Technology Transfer (STTT)*, Vol. 9, Issue 2, pp.179 – 192.

Damm, W. Josko, B., Pnueli, A. and Votintseva, A. (2002). Understanding UML: A Formal Semantics of Concurrency and Communication in Real-Time UML. *First International Symposium, FMCO 2002*, Leiden, The Netherlands, November 5-8, 2002, Revised Lectures, pp. 71-98.

DiPippo, L. C. (2000). A UML Package for Specifying Real-Time Objects, *Computer Standards & Interfaces*, Volume 22 , Issue 5 (December) pp. 307–321.

Douglass, B. P. (1998), *Real-Time UML*, Addison-Wesley, Reading, MA.

Dubinin, V. and Vyatkin, V. (2004). UML-FB – A Language for Modeling and Implementation of Industrial-Process Measurement and Control Systems on the Basis of IEC 61499 Standard, *Proc. of the 6-th International Conference of Science and Technology “New Information Technologies and Systems (NITS’2004)*, Penza, Russia, June 17-19, Part 2, pp.77-83.

Estefan, A. J. (2008). Survey of Model-Based Systems Engineering (MBSE) Methodologies, INCOSE MBSE Focus Group.

Gerard, S., Terrier, F., and Tanguy Y. (2002). Using the Model Paradigm for Real-Time Systems Development: ACCORD/UML, *Proc. Conf. Advances in Object-Oriented Information Systems, OOIS 2002 Workshops*, J.-M. Bruel and Z. Bellahsene, eds., pp.260-269.

Graf S. and Ober I. (2003). A real-time profile for UML and how to adapt it to SDL. In: Proceedings of SDL Forum 2003, http://citeseer.ist.psu.edu/graf03realtime.html

IEC Technical Committee TC65/WG6, IEC-61499 Industrial Process Measurement and Control Specification, IEC Draft 2000.

Kent, S. (2002). Model Driven Engineering. *In Proceedings of IFM2002*, LNCS 2335, Springer.

Lewis, R. (2001). Modeling control systems using IEC 61499, The Institution of Electrical Engineers, London, United Kingdom.

Martin, G., Lavagno, L. and Louis-Guerin, J. (2001). Embedded UML: a merger of real-time UML and co-design, http://www.gigascale.org/pubs/101.html

Medina, J. L., Drake, J. M. and Harbour, M. G. (2001). UML-MAST: Modeling and Analysis Methodology for Real-Time Systems Developed with UML CASE Tools, Proceedings of 13th Euromicro Conference on Real-Time Systems, Delft, Netherlands, IEEE Computer Society Press, pp.125-134, June.

Meunier, J.-N., Lippert, F. and Jadhav, R. (2003). RT modeling with UML for safety critical applications: the HIDOORS project example, Proceedings of SVERTS 2003, Co-located with UML 2003, San Francisco, CA, U.S.A., October.

Mwaluseke, G. W. and Bowen, J. P. (2001). UML Formalisation, Literature Survey, 6 September.

OMG-MARTE (2006). Object Management Group. UML Profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE). Request For Proposals.

OMG-MDA (2003). MDA Guide version 1.0.1. OMG document omg/2003-06-01, 2003.

OMG-QoS (2004). UML Profile for Modelling Quality of Service and Fault Tolerance Characteristics and Mechanisms. OMG Adopted Specification, ptc/2004-06-01, June.

OMG-SysML (2006). The OMG Systems Modeling Language, http://omgsysml.org/index.htm, May.

OMG-SPT (2003). UML Profile for Schedulability, Performance, and Time Specification. Version 1.0, formal/03-09-01, Sept.

OMG-UML (2010). OMG. http://www.omg.org/

OMG-UML-Superstructure (2004). UML 2.0 Superstructure Specification. OMG Revised Final Adopted Specification (ptc/04-10-02), October.

Selic, B. and Rumbaugh, J. (1998). Using UML for Modelling Complex Real-Time Systems, Rational Software Corporation.

Selic, B. (2003). The Real-Time UML Standard: Definition and Application, Rational Software.

Thramboulidis, K. C. (2004). Using UML in Control and Automation: A Model Driven Approach, *2nd IEEE International Conference on Industrial Informatics INDIN’04*, 24th -26th June.

Verschaeve, K. (2001). UML - SDL round-trip engineering through incremental translation of changes, Vrije Universiteit Brussel, Faculty of Science, February.