Empirical Evaluation of SysML through the Modeling of an Industrial Automation Unit

Marcos Vinicius Linhares, Alexandre José da Silva, Rômulo Silva de Oliveira
Federal University of Santa Catarina
Campus Universitário – Trindade
CEP:88040-900, Florianópolis, Brazil
marcos, ajs, romulo@das.ufsc.br

Abstract

Industrial automation systems may include people, hardware, software and others necessaries to produce the desirable results. The SysML modeling language is being proposed, by OMG and INCOSE, as a language that allows the system description correctly and consistently among various participants of the same project (software, mechanical, electrical and others engineers). The objective of this work is to evaluate the SysML proposal as a description language for industrial systems through the modeling of an industrial automation experimental unit. A brief system description of the case study will be made, the SysML diagrams will be presented during the system modeling and finally the comments and considerations about the models and the modeling task.

1 Introduction

A system is a construction or collection of different elements that together they produce results that cannot be obtained by each one of its elements in separate. The elements, or their subparts, can include people, hardware, software and other necessary ways to produce the desired results [6].

There are several proposals that address the use of UML (Unified Modeling Language) for systems engineering [1][2]. Each approach has its own deficiencies. Sometimes they create separate models that are difficult to integrate. Some of them break up with the standardized characteristics of UML.

Those that know UML find it to be an efficient modeling language, but its roots are firmly fixed in the software modeling area. The Object Management Group (OMG), responsible for maintaining the UML language, establishes that the “UML is a visual and general purpose language used to specify, to visualize and to document models of software systems [9].” However, many system engineers believe that UML is sufficiently flexible and robust to support extensions and to address its use for the needs of their specific domains. One of its strong points is the UML specialization mechanism that allows several applications, with their own domain characteristics, can be used together with UML profiles (encapsulating terminologies and specific substructures of a specific domain).

It is believed that UML could potentially be a modeling language for system engineers, contemplating analysis, design and verification of complex systems, intending to enhance system qualities, increasing the capacity to exchange information among many engineering tools and to help to fill the gap between systems and software engineers [3].

In 2003, at the same time that UML 2.0 was being defined, OMG launched a RFP (Request For Proposals) for a definition of a language similar to UML for systems engineering. This RFP was developed together by OMG and the International Council on Systems Engineering (INCOSE).

In agreement with the RFP, this new language should support the specification, analysis, design and verification of complex systems [4]. It should be done by capturing information of the system in a precise and efficient way, facilitating integration and reused in a larger context; analyzing and evaluating the specified systems, to identify and to solve requirements of the system, to distribute projects and to support exchange among them; communicating the information of the systems, correctly and consistently among several participants of the same project (software, mechanical, electrical and other engineers).

In response to this RFP, it was created the Systems Modeling Language (SysML) [7]. The objective of this work is to empirically evaluate the proposal of SysML as a language for description of industrial systems through the modeling of an experimental industrial automation unit. There are some SysML draft documents, but in this paper we use, at this moment, the latest version (1.0) which is being evaluated by the Analysis and Design Task Force (ADTF) group to become the adoption process by the OMG. It was renamed as OMG SysML [5].

The objective of this paper is to do a practical evaluation of the modeling language SysML, in the sense of
identifying its capacities and its limitations for the modeling of industrial automation systems. It will be used as case study an experimental automation unit system. In section 2, the SysML language is presented. In section 3, the experimental automation unit system case study is described. In section 4 the system is modeled using the SysML diagrams. The section 5, contains comments about the experience with SysML in the modeling of an industrial automation system. Section 6, has the final considerations.

2 The Systems Modeling Language (SysML)

SysML is designed to provide simple but powerful constructions for the modeling of a great variety of system engineering problems. It reuses a subset of the UML 2.0 models. This subset is called UML4SysML. Since that some models were modified, it actually a profile of UML for SysML. It also provides additional constructions, with the objective of satisfying the needs of system engineering and the requirements of OMG.

It is particularly effective for specifying structure, behavior, requirements and constraints on properties of the system to support analysis. SysML should be supported by two evolving interoperability standards, OMG XMI 2.1 (standard for the exchange of information among modeling tools using UML 2.0) and ISO AP233 (standard for the exchange of information among engineering tools).

Figure 1 shows the modifications in the diagrams reused from UML 2.0 as well as the new diagrams of SysML. The specification of SysML is classified in three basic types of models: the structure models, the behavior models and the requirement models. For each one there are defined constructions that are used in a specific model. Some constructions can be used together with several model types. The constructions, as well as the models, are also classified in three types: the structural, the behavioral and the cross-cutting constructions.

![SysML Diagrams](image)

The behavioral constructions specify the dynamic parts used in SysML. The diagrams that include the structural constructions are: Package Diagram, Block Definition Diagram, Internal Block Diagram and the Parametric Diagram.

Some structural generic elements are: the model elements (rebuild the core of UML 2.0 packages and include extensions to provide some basic capabilities and models management); the blocks (reuse and improve the class structure of UML 2.0 to provide the basic capability of describing the decomposition and interconnection of the system, and also different types of system properties); the ports and flows (contain the semantics to define how blocks are extended to be used together with ports and flows); and the constraint blocks (define how blocks are extended to be used with the Parametric Diagram that models the constraint network on system properties, that is to give support to the reliability analysis and others analysis).

The behavioral constructions specify the dynamic parts used in the behavior diagrams of SysML, including among them: the Activity Diagram (used to describe the control flow and the input and output flow among the actions), the Sequence Diagram, the State Machine Diagram and the Use Case Diagram, the same ones used in UML with little or any modification. The behavior constructions used in the diagrams are divided in: activities (basically, the same ones defined in UML 2.0 with some extensions to allow continuous elements as activity parts); interactions (where it is defined the constructions for the behavior based on messages used in the Sequence Diagram); State Machines (used to describe the behavior of a system based on its states and its transitions); and Use Cases (they describe the behavior and the use of a system in terms of its high level functionality, like in UML).

3 The Case Study

In the Systems and Automation Department at Federal University of Santa Catarina (UFSC) there is an experimental unit of industrial automation that is used to demonstrate the operation of several control strategies using the same equipments and supervision tools. Figure 2 illustrates the experimental unit.

The experimental unit uses Foundation Fieldbus as the communication network and it allows the implementation of applications controlling the flow, level and temperature of a fluid. The unit is composed by several intelligent devices, and a PLC (Programmable Logical Controller), integrated in the network by a universal bridge that allows the connection of an Ethernet network to a Foundation Fieldbus. It also has a station for operation and supervision of the unit, constituted of a computer and supervision software. This software receives data acquired in the plant and presents it on the computer screen. That integrates the supervision and the control levels of the unit.

The intelligent devices based on the Foundation Fieldbus technology are capable of executing, in a distributed
environment, all the necessary control strategies for an industrial application. As a whole, the experimental unit has 9 intelligent devices: 2 valves, 4 pressure devices (2 to measure level and 2 to measure flow), 2 temperature sensors devices and 1 current device. The experimental unit is complemented by other equipments such as frequency inverters, resistances, temperature transmitters, valves solenoids, optical level sensor, pumps and others.

4 The SysML Modeling

The experimental unit was used to evaluate SysML modeling. SysML models describe its structure, behavior and requirements from the point of view of the system engineer.

The Block Definition Diagram (BDD) describes the structure of the system or of a subsystem by using the block as its basic unit. The relationships among them (composition, inheritance, aggregation and others) and the diagram format are the same ones used in the Class Diagram of UML 2.0. Figure 3 illustrates the Block Definition Diagram for the experimental unit system. It is composed by three subsystems. The experimental unit was divided in: physical subsystem, control subsystem and supervision subsystem. This BDD also shows how each subsystem is connected with each other. For example, the control subsystem uses the smart devices to control the physical subsystem.

The physical subsystem is the part of the system which shows the physical structure in terms of installed equipment. The block definition diagram shown by Figure 4 illustrates this subsystem. It is composed by different types of equipments such as valves, transducers, tanks and the devices that connect the physical subsystem to the control subsystem.

The control subsystem contains the parts of the system responsible for the execution of the control strategies. It includes the Foundation Fieldbus network, the PLC and the universal bridge that connects the controlled plant to the supervision system. The supervision subsystem is responsible for verifying the operation of the control strategy implemented through the data acquisition from the field devices and the presentation of these to the user in a graphical form.

The Internal Block Diagram (IBD) describes the internal structure of a block, its parts and the connections among them. The connections are made through the use of flow ports, where matter, energy or data may flow. Service ports connect the services provided and/or requested by a certain block. Figure 5 shows how the equipments are installed in the experimental unit. The flow of several elements can be seen in the system, including both the direction of the flow and the specification of what flows. For example, FS\_FLUID specifies a fluid (flow specification) that passes for several elements of the system. It can be anything from water to some petroleum by-product.

Figure 6 shows the IBD of the control subsystem. It describes the interconnection of several devices of the plant in a Foundation Fieldbus. It can be noticed the existence of the flow specifications that describe what is flowing among the equipments installed in the network. Figure 7 shows how the devices are connected with the supervision station. It presents a service view, that is, a logical view of the system. In this case the used ports are service ports and not flow ports. Since they are reused from UML, they include all the associated elements, such as the requested interfaces and the provided interfaces by the specified services.

In order to support analysis tasks, the Parametric Diagram models a network of constraints on the properties of the system. It can be used to identify performance parameters, relationships among parameters and mathematical formulas, static values or other parameters that integrate the network of constraints. The pressure device, showed in the Fig. 8 has some function blocks like AI (analog input functions), DSP (display functions), ARTH (arithmetic functions) and others. The relationship between the parametric constraints of the ARTH block can be seen in Fig. 9. It illustrates the various types of arithmetic equations used in this function block. In order to know how the arithmetic function performs its operations it is necessary
to have a more detailed view of what is performed by a parametric diagram illustrated in Fig. 10.

The objective of the Use Case Diagram (UC) is to show, in a simplified way, the behavior of a system to an agent that is external to the system. It presents the system from the user’s perspective, including functions, services and identifying which user is related to them. This diagram is used to describe how the system is used. Figure 11 shows the unit from the point of view of its operation. Of all the diagrams, the Use Case is the most abstract, flexible and informal. It serves as the base for other diagrams, specially the sequence diagram.

In order to describe the control flow between the users and the system or between parts of the system is used the Sequence Diagram (SD). It identifies the sequence in which the events happen in a certain process or case. It shows the elements involved in the dispatch of the events and the order in which they are dispatched. This diagram is based on the use case diagram, usually a sequence diagram for each use case. Obviously the sequence diagram is also related with the block definition diagram since these are the interaction elements of the system.
12 shows the case of plant operation.

The State Machine Diagram (STM) is used to show the discrete behavior of the experimental unit through finite state transitions (Fig. 13). This diagram represents the operation of the plant in terms of its transitions and states. The activities of the system can be either continuous or discrete and they are represented by their states. When the activities are continuous it is necessary to use the activity diagram to represent these in a subsumed state machine.
The Activity Diagram (ACT) emphasizes the inputs and outputs, sequences and conditions. It provides a flexible connection between the blocks and their behaviors. This diagram was extended to better support the Functional Block Diagrams (EFFBD) and their continuous and discrete control flows. Modeling continuous dynamics is important in most complex systems where hardware and software integration exist. Figure 14 shows the activities for the level control strategy. This is an example of control strategy that can be implemented in the experimental unit.

The Requirement Diagram (REQ) specifies the capability or condition that should be satisfied, traced and/or allocated by the system. The requirements are text based constructions related to system model elements. Figure 15 shows the requirements for the level control strategy. It is possible to know which element (block, another diagram, etc.) satisfies the requirement and some others information about the systems behavior may be traced by this diagram.

The allocation is used by the system engineers to map elements, activities or other structures of an user model into a cross-association that shows where are the relationships between each one. The allocation can be represented by a cross-referenced element model into another diagram (like block definition diagram or internal block diagram) or in an allocation sparse matrix.

From the activity diagram, show in Fig. 14, it is possible to generate an allocation matrix (Table 1) for the device elements. This table, in this case, is intended to show where the activities are in each specific device in a level control strategy.

5 Comments on SysML Models

Since SysML is a UML 2.0 subset and some diagrams are imported without modification, it was necessary the use of support material on UML during the case study modeling. But, the system modeling task is quite differ-
ent form an object-oriented software modeling, and for this purpose some diagrams were modified to show system characteristics.

In UML is common to listen that “20% UML diagrams made 80% UML modelling work”, in SysML diagrams it is not be do, each diagram is interrelated with another performing an hierarchical structure that shows what the systems does and how the systems work.

The most useful diagram depends on what the system engineer wants to show and where are seeing the diagrams. But, the new and modified diagrams have some strengths and weaknesses that must be analyzed.

The block definition diagram and the internal block diagram describe the external and internal system structure, respectively. They allow the description of which elements are interconnected and how they are interconnected. They also allow a top-down and/or a bottom-up modeling giving an abstract/concrete system level to different system elements. Unfortunately, to describe a complex system it is important a greater organization because depending on the system under modeling, and the view that wants to show, the number of BDD and IBD diagrams and model elements increases very quickly.

The requirement diagram and the parametric diagram are developed for systems engineer needs and they present an important role during the system modeling. Initially, the requirement diagram was perceived as a great improvement over UML models, because it shows how the requirements are satisfied by the system elements. That requirements may be provided by the user, the environment, or by the system itself. Even so, it presents a high degree of informality. Requirements are described through text based on natural language. It may insert inconsistencies during the extraction of information from the diagram. However, requirements can be grouped in a clear way, also documenting their origin (standards or more detailed specifications) or tracing their destination. The requirements also can be used to provide useful acceptance tests before the system deployment.

The parametric diagram is used to model mathematical dynamics involved in the system. Since those are specific application systems, they present calculations for control and processing of signals that, once modeled, they aid in the definition of routines with timing constraints.

The activity diagram had little modifications to support continuous elements that are present in many systems types and it is an important element to the system engineer. But, the activity diagram are modified from the UML 2.0 to support some verification (now it contain some elements that looks like a Petri Net) and this new model element introduces another challenge, how to verify discrete and continuous behavior together.

The state machine diagram and also the sequence diagram provide the same structures models that are used in the UML models and they are addressed, respectively, to discrete behavior and message passing behavior.

The use case diagram and the package diagram are the same as in UML. In a similar way to the design of software systems, these diagrams aid the system modeling during its initial stage, where the level of abstraction of the system is high.

The use of some models is not very intuitive. This is justified by the fact that the same diagram can represent different views of the same model, depending on the elements that are used. Depending on the size of the system, the number of diagrams grows very quickly when one increases the level of detail. In a very complex system can be required a more abstract level to represent the system elements and other domain models (software, electrical, mechanical models) can be used to represent the detailed elements. In this viewpoint the SysML is a powerful language to integrate a great variety of domains in a first modeling stage that can be used to system development integration and control.

6 Final Considerations

In this paper SysML was evaluated as a proposal for the modeling of industrial automation systems. In spite of presenting a steep learning curve, the language allows the building of diagrams that easily represent the structure of the system that is being described. Through its diagrams it is possible to have an idea of the final system.

The biggest difficulty faced while using SysML was not the fact of it being in a preliminary version. The biggest problem is that during a time period of 6 months the specification was changed four times. The last time it was the beginning of April 2006, when version 1.0 was introduced. This last one seems to be stable and the one that will probably be refined until becoming a standard adopted by OMG.

The use of some diagrams is not very intuitive. The fact that it is possible, in the same diagram, to represent different views of the same model, depending on the elements that are used, does not help the system engineer. Depending on the size of the system, the number of diagrams grows very quickly when one increases the level of detail. Parece que tem algum coisa parecida logo acima. Qual fica?

Due to the fact that several diagrams were imported from UML 2.0, many of the models make reference to the UML specification, including diagrams that were modified. That makes the modeling work more difficult for
those that do not have a previously knowledge of UML. The specification of SysML is a small document when compared to that of UML. That gives to the system engineer the illusion that the work is simple.

For this paper we opted for a more generic view of the system and of the physical subsystem, including some new SysML diagrams in order to evaluate them. This choice was motivated by the short available space to expose the plant modeling. The consequence was that several diagrams that would bring a deeper view of the system were left out of the paper. Anyway the included diagrams allow a preliminary evaluation of the potential of SysML. The language is really powerful and to this moment it has supported the necessities of our modeling. For the system engineer, it is a language easier to understand when he/she tries to read it than when he/she has to write models with it. In other words, the review of the diagrams for other engineers won’t demand the same training level that the system engineer needs to draw the diagrams.

In spite of the problems, the result is positive. SysML allows the system engineer to describe the system, including hardware and software aspects simultaneously with the other elements of the industrial automation unit. However, SysML modeling requires the project team, generally formed by engineers, to have knowledge of UML. This requirement could be a barrier to its acceptance in a large scale.

For software developers that are used to UML 2.0, the use of SysML is straightforward. A SysML specification would be a much better start for the system development than a specification in natural language or, still worse, several electrical and mechanical diagrams.

Since the system development team is multidisciplinary, including professionals from different domains, SysML is a very interesting proposal to improve the communication among them. To illustrate such statement we consider a meeting on certain system elements. An electronic engineer has electronic diagrams, a control engineer has control diagrams and a software engineer has software diagrams. The interaction among the team is minimum, because one does not have knowledge of the modeling language used by the other. However, if SysML was used, the modeling language would be the same for all the persons with different backgrounds. It would improve the interaction among the team and decrease the time to market of a system.

References