Metamodelling of production systems process models using UML stereotypes

Hervé Panetto, Jean-François Pétin

Metamodelling of production systems process models using UML stereotypes

Hervé Panetto, Jean-François Pétin

CRAN CNRS UMR 7039/University Henry Poincaré Nancy I, F54506 Vandoeuvre-les-Nancy, France

E-Mail : {Herve.Panetto, Jean-Francois.Petin}@cran.uhp-nancy.Fr

Biographical notes:

Hervé PANETTO, Associate Professor, has strong experience in Enterprise Systems modelling and meta-modelling. He is expert in AFNOR (French National standardisation body) and ISO TC184/SC4. He has responsibilities on different European projects including IMS Smart-fm IST project and the UEML IST European Thematic Network. He is core member in the INTEROP NoE (Network of Excellence on Interoperability Research for Networked Enterprises Applications and Software) of the European 6th FP. He is currently Vice-Chair of the IFAC Technical Committee 5.3 "Enterprise integration and networking".

Jean-François PETIN, Associate Professor, develops research activities on formal models and System engineering approaches for global satisfaction of enterprise requirements. He has been consultant for the French EDF power supplier in formal system modelling with the B method. He manages several tasks on these subjects in European projects (PRIAM, IAM-Pilot). He has been also involved in the EP 21955 "IMS – Working group". He participates to the UEML European Thematic Network.

Metamodelling of production systems process models using UML stereotypes

Abstract: This paper contributes to a formal framework for the automation of production systems. Automatic control systems deal currently with theoretical modelling techniques aiming to formally define the behaviour of a control system when process behaviours and system requirements are well defined. Our approach aims to facilitate a common and consensual understanding of an automated system at the early stage of a systems engineering process. This approach is based on the use UML stereotypes, based on a systemic approach, to globally formalise a production system satisfying the following predicate: Control Systems Requirements $\wedge$ Process Systems Requirements $\Rightarrow$ Production System Requirements.

Keywords: process modelling, metamodelling, UML stereotype, modelling contraints

1 INTRODUCTION

The formulation of the automatic synthesis of a control system, as proposed by [1], consists in defining the (unknown) control rules of the (known) dynamics of a physical system, starting from the behavioural (known) goals to be met, while satisfying the following condition:

$$Control\ Rules \wedge Dynamics \supset Goal \qquad (1)$$

In this way, approaches such as the promising Supervisory Control Theory [2][3] deal more or less with automatic synthesis of control system from dynamical properties of process system and system requirements.

Control engineering cannot be in practice restricted to dynamical issues in order to master the global complexity of an industrial system. In particular, emergent technologies (Internet, PLC Web server, MEMS, …) stress the role of information and communication within the control applications by considering the control devices as data servers for enterprise systems such as Manufacturing Execution Systems (MES) or Enterprise Resource Planning (ERP). Extension of the Fusaoka's predicate to

$$Process\ Specifications \wedge Control\ Specifications \supset System\ Specifications \qquad (2)$$

covers this evolution by considering a global specification of the services required for manufacturing control, and their realisation through process and control systems to be designed [4]. The aim is to facilitate a common

and consensual understanding of an automated system which can be shared among the various engineering actors involved in automation life cycle and to postpone the use of skill-oriented models suitable to design more precisely each of the system processes

This paper presents a unified modelling approach, based on UML stereotypes [5], that helps in establishing the above automation predicate (2) ensuring the global consistency of the system model [6].

Second section states the meaning of systems in the design process from the classic mathematical vision in automatic control to the industrial pragmatic vision in systems engineering. Our approach for formalising the use of UML in the specification of a production system is presented in section three and detailed and illustrated in section four using a case study. Benefits, limits and future extensions of the work are discussed in the conclusion.

2 "SYSTEM" CONCERNS AND ISSUES FOR AUTOMATION ENGINEERING

2.1 *From automatic control …*

The Fusaoka's formulation (1) is compliant with the modelling process of control system as addressed by what is currently called "system theory" in automatic control [7]. The modelling process consists in five steps: (a) modelling and analysis to understand how an existing system actually works, (b) design and synthesis to build a system that behaves under conditions according to the desired goal, (c) control to mathematically design the control rules as efficiently as possible, (d) performance evaluation to measure how the designed and controlled system really satisfies the performance objectives, and (e) optimisation to tend to best possible behaviours.

The two first steps refer to a more or less informal (qualitative) modelling process. The system is an intuitive model of something real (a quantity of material, energy, information… a robot, a cell, a factory …) to be controlled for performing end-user goals (see Figure 1). The modeller's intuition remains important to build the model as an abstraction of the real system by identifying the appropriate input, output and state variables in order to logically define the desired system behaviour. The three last steps refer to a formal (quantitative) modelling process consisting in determining suitable mathematical relationships involving the system dynamics.

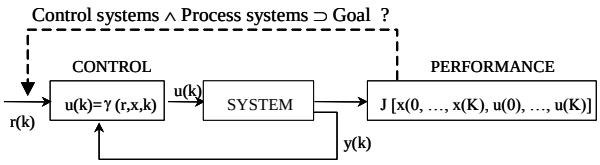

Figure 1. A closed-loop system with dynamical performance optimisation

Based on these theoretical foundations, the concept of control synthesis has been introduced [2] to support automatic definition of a control model from formal models of the process dynamics and expected behaviour. Synthesis algorithms are therefore an answer to Fusaoka's predicate since they generate control rules which constraint the behaviours of the automated system (process and control parts) with regards to the forbidden states of the goals specification.

2.2 *… to systems engineering*

However, even if these techniques have demonstrated their efficiency in control design, two key limitations for a use in systems engineering[1] must be pointed out: the scope of the synthesis models and the project life cycle.

Control theory and synthesis techniques are unanimously recognized as limited to the dynamics aspects. Information and communication issues are not considered within the scope of these theoretical approaches. This point is illustrated in the Figure 1 by the term "system" that remains restricted to the early representation of the real part ("thing") to be controlled. It does not appoint the final assembly of all interconnected components within a manufacturing system, such as the process parts, the actuation and measurement parts, the control parts, the information and communication parts, the human-machine interfaces… which constitutes the realistic purpose of automation engineering. Note also that the performance evaluation and optimisation refer, in an open-loop, to the intrinsic properties of the control model, rather than, in a closed-loop, to the extrinsic properties the final system is intended to satisfy.

Control theory and synthesis techniques are also recognized to rely on mathematical models of the process behaviours and system requirements that are more or less considered as given inputs of the control design activities. According to [8], automation engineering, even if purely mathematically modelled, reflects in fact the heuristic capabilities of a collaborative process, which more or less transforms end-user's intentional expectations into a delegated artificial artefact aiming to manufacture final products and services. Pointing out this vi-

tal cognitive role (see Figure 2) of the design activity leads to extend the principles of cybernetics towards engineering and manufacturing in order to better take into account human causations (intention, intuition, interpretation, information, networking, …) beyond physical ones (material, energy, data, communication, …). Major consequences on the various models elaborated during an automation project are that they must be able to be "dynamic" i.e. to evolve from an abstract representation to a more concrete one. As an example, it is not unbelievable to consider a control model that is more or less deterministic even if implementation of control rules needs to be strictly deterministic: it only means that the abstract model is progressively refined by adding concrete details which constraint some deterministic choices.

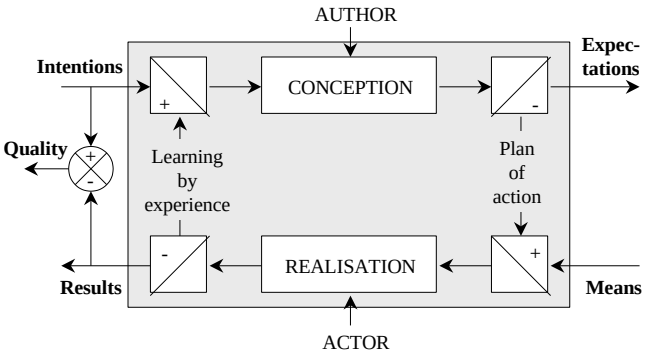

Figure 2. Cybernetic cognitive model of the design activity [8]

One key problem of this cognitive design activity remains the duality between the realisation of the intentions by means of skill-oriented technologies (*Control systems ∧ Process systems)* and the transmission of the intentions meaning by means of computerised technical representations through all the actors involved in industrial systems automation, with regard to the end-user's expectations (⊃ *System Requirements)* [9]. In order to cope with this boundary between informal (meaning) and technical representations (means) in design, automation engineering needs to handle abstract models that offer a consistent vision of the various aspects of an automated system (functionalities, behaviour, information, communication, …) among the various engineering processes (control engineering, mechanical engineering, …) before using skill-oriented representations. These requirements can be covered by approaches such as system theory through methodological guidelines or by the UML through a specific profile.

[1] INternational COuncil on Systems Engineering (INCOSE), http://www.incose.org/

System theory proposes generic guidelines for the modelling of a system within its environment. Particular attention is paid for the representation of interactions between components that may lead to emergent properties that cannot be satisfied by any of the individual parts. From an engineering point of view, this theory is supported by various methods and tools, such as the SAGACE method that tries to provide a consistent set of models covering three major points of view of a system : functions (what the system has to do), structure (what the system is) and evolution (system behaviour).

3 FORMALISING THE USE OF UML

The Unified Modeling Language (UML), an OMG standard, is a 9 notations language that results from the unification of object-oriented approaches. Its notation includes the following kinds of diagram for modelling different perspectives of an application: use case diagrams for capturing the users 'requirements; class diagrams, and object diagrams for specifying static relationships between objects; state transition diagrams, activity diagrams, sequence diagrams and collaboration diagrams for describing inter or intra-objects behaviour; component diagrams and deployment diagrams for specifying the implementation constraints; the Object Constraint Language (OCL), a simple query and navigation language for defining the well-formedness rules of UML models. Even if UML provides a wide notations toolbox, it does really not cope with our needs for system modelling. Indeed, UML deals with descriptive notations without considering any method for establishing the models and more particularly for enabling an incremental specification method based on a progressive refinement of the different models. Although UML is a general purpose modelling language, it contains extensibility mechanisms that can be used to tailor it to specific domains. These extensibility mechanisms can be understood as indirect modification, at the model level, of the UML meta-model. The standard extensibility mechanisms of UML are stereotypes, tagged values and constraints. These extensibility mechanisms are called "lightweight extensibility mechanisms" in contrast to the direct manipulation of the UML "meta-model" that can be interpreted as "heavyweight extensibility mechanisms" (addition of new meta-classes, meta-associations, etc.).

In order to give support to the gradual adoption of "standard" UML extensions, OMG has introduced the concept of "UML profile" which, in spite of the lack of a normative definition, has already been used in several OMG technical groups. A "profile" might be defined as a "specification that specialises one or several

standard meta-models, called “reference meta-models”. In the context of OMG, all those reference meta-models must be compliant with the meta-meta-model prescribed by MOF (Meta-Object Facility) [10] of OMG. These profiles are normally intended to customize UML for a particular domain or purpose. They can also contain data types that are used by tag definitions for informally declaring the types of the values that can be associated with tag definitions. Indeed, these extension mechanisms are a mean for refining the standard semantics of UML and do not support arbitrary semantic extension. They allow the modeller to add new modelling elements to UML for use in creating UML models for process-specific domains such as enterprise models . Moreover, as the UML specification relies on the use of well-formedness rules to express constraints on model elements, this profile uses the same approach. The constraints applicable to the profile are added to the ones of the stereotyped base model elements, which cannot be changed. Constraints attached to a stereotype must be observed by all model elements branded by that stereotype. If the rules are specified formally in a profile (for example, by using the Object Constraint Language (OCL) for the expression of constraints), then a modelling tool may be able to interpret the rules and aids the modeller in enforcing them when applying the profile. UML is currently used to define common semantics to the existing various frameworks [11].

Our objective is to propose a unified modelling method that completes the UML by providing a reasoning framework based on the automation predicate (2).

Next section presents a theoretical background about process specification. The last section specifies some of the UML stereotypes that are the candidates for supporting our approach because of their usefulness for capturing the various system specifications.

4 FORMAL SPECIFICATION OF THE PHYSICAL PROCESS

Recent works in the area of complex physical system modelling [12][13] have highlighted the benefit of using object oriented representations to mix theoretical modelling with identification techniques when necessary. In this context, a physical system is modelled as a network of interconnected processors acting as white boxes in the case of theoretical modelling or black boxes in the case of identification. In order to facilitate their reusability, most of these processors are described independently from each other and without any causality constraints. However, when building a physical system representation by connecting these independent processors, the modeller has to take care about:

– the relationships between input and output flows of each elementary processors which may be constrained by some physical rules of conservation (energy or flow balance, specific features of the transformations, …),
– The connection between elementary processors which are physically limited to some enabled configurations (causality relationships between physical variables).

In the context of our study, the key issue we want to address is the correctness of these relationships, with regards to physical laws, of the process structure provided by object-oriented representations. Our approach is based on the formalisation of some expert knowledge in the area of physical systems modelling that help in re-using the formalised knowledge. More precisely, our work refers to the Paynter's classification of the physical variable [14] and its application within a systemic approach proposed by Feliot [15]. Let's have a quick look on these results before detailing our contribution.

4.1.1 *Theoretical foundations*

Physical variables have been classified by Paynter according to four basic sets: effort, flow, impulse and displacement. Applied to hydraulic systems (related to our case study), the following variables: pressure, fluid flow, volume and moment of pressure can be assumed to respectively belong to Effort, Flow, Displacement and Impulse sets. Paynter's tetrahedron of states (see Figure 3) provides the relationships between those variables.

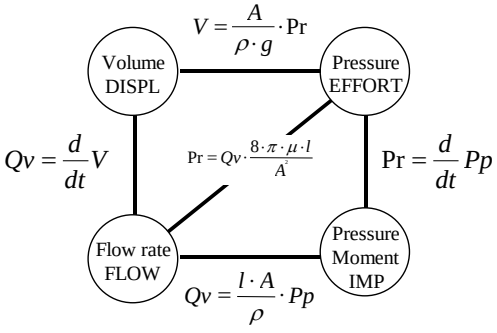

Figure 3. Formal typology of state variables for physical systems

Considering that some of those physical variables are tightly coupled (we cannot speak about tension without intensity or fluid flow without pressure), Paynter has proposed to consider three main couples: (effort, flow) known as a power, (effort, displacement) known as a potential energy and (impulse, flow) known as a

kinetic energy. According to these definitions, a physical process is defined as the transformation between several couples. It results three kind of physical processes : from power to power (P/P) with proportional relationship, from energy to power (E/P) with a derivative relationship and from power to energy (P/E) with an integral relationship.

Feliot has formally demonstrated that these three kinds of processors are strictly equivalent to the three basic processors proposed by the system theory: (P/P), (P/E) and (E/P) processors are respectively equivalent to Shape, Time and Space systemic operators. From the process modelling point of view, Shape, Time and Space operators respectively support the physical transformations of a product, its storage and its transport. On the basis of system theory, Feliot has proposed some structuring rules for interconnecting these operators.

Our contribution consists in formalising these theoretical foundations using UML stereotypes and OCL well-formedness rules with two objectives:

- to provide basic constructs for the modelling of process systems, and more particularly in the context of our case study for the modelling of hydraulic systems,
- to provide a formal framework for the modelling of a particular process system using the instantiation of the basic constructs.

4.1.2 *Formalisation of basic constructs for process modelling*

In a first level of abstraction, the basic systemic processors are formalised within UML stereotypes without taking into account a precise description of the dynamics of the supported transformations. This leads to define three basic constructs that include: a definition of the variable couples involved in the transformation, the invariant relationship that link the two variables inside each couple ; these relationship are those defined by the Paynter's tetrahedron of states.

A very abstract description of the transformation - by saying that any operation that maintains the previous invariant - will be considered as possibly done.

The chosen case study (see Figure 4) is a simplified part of our laboratory platform dedicated to Intelligent Manufacturing Systems.

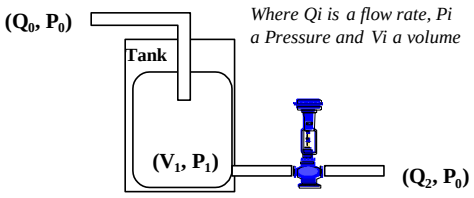

Figure 4. The case study

The specification focuses on an automated system aiming to deliver a potential energy by controlling the water level inside a tank.

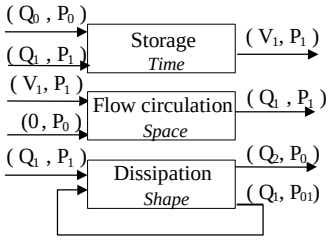

Figure 5. B formal typology of physical processors

Figure 5 presents a Time processors acting in the area of hydraulic systems where (Q0, P0) and (Q1,P1) represent respectively the input and output fluid of the storage system with two couples (flow, pressure) while (V1,P1) represents the volume and pressure within the tank. In the same way, Shape processor has been formalised with (Flow, Pressure) as input and another (Flow, Pressure) as output by taking into account the problem of energy dissipation while Space processor is defined as transforming (Volume, Pressure) and (Flow, Pressure) into (Flow, Pressure).

Note that all the relationships described within the invariant and the operation, involve some parameters such as A (tube or tank section). It clearly appears that some of them should be further continuously modified by the control system. For the moment, these parameters are considered as constants. Indeed, the choice of the variables which will be considered as controllable (actuators) or observable (sensors and transmitters) is not strictly relevant from the process modelling.

In our context, a physical system is supposed to be modelled as a network of interconnected processes. Such a representation of our case study is presented in the Figure 6: The water storage within the tank is modelled by a Time processor while the valve is modelled both through a Shape processor (energy dissipation) and a Space processor (water transportation).

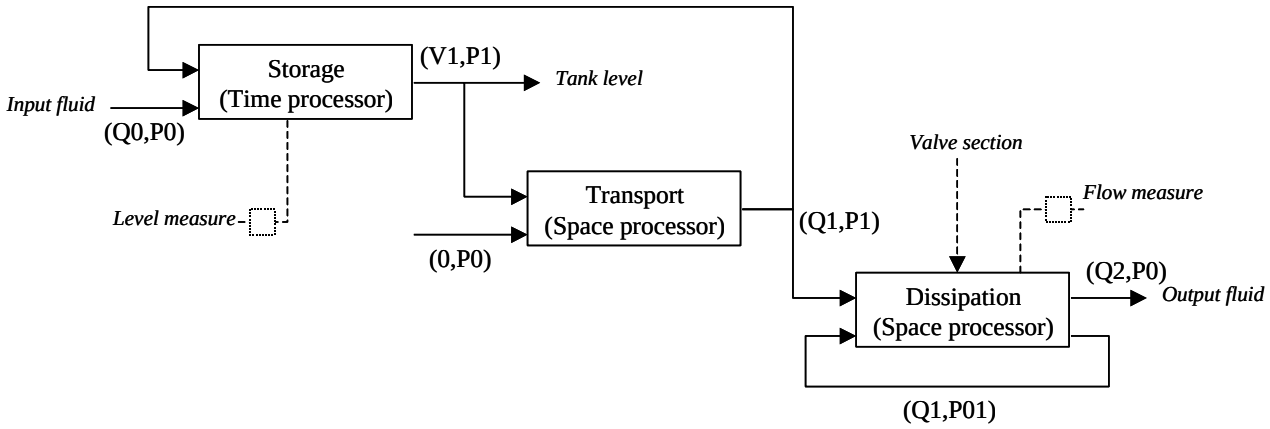

Figure 6. Network of basic processors

5 UML STEREOTYPES FOR PRODUCTION SYSTEMS

As we have shown in §4.1.1, our approach relies in three types of processors: Space, Time and Shape. Figure 7 shows the three stereotypes defining these three types of processors. These stereotypes are based on the UML Classifier metaclass. In that way, as specified by the UML metamodel, they have, at least, structural and behavioural features such as attributes and operations, and they are linked together by associations.

The UML association between two processors is a Flow which is subtyped as MaterialFlow, Information-Flow or ResourceFlow by the definition of three stereotypes based on the UML Association metaclass (Figure 8).

The flows relate, as defined by Paynter (§4.1.1), on PhysicsTypes specialised as Power, Potential Energy, Kinetic Energy. Each one is defined by a couple of variables which belong to Effort, Flow, Displacement and Impulse sets. The PhysicsType stereotype is based on the UML Classifier metaclass.

In order to cope with our context of modelling production systems, our stereotypes have to deal with some constraints, based on physical laws as stated by Paynter, specified by well-formedness rules written in OCL.

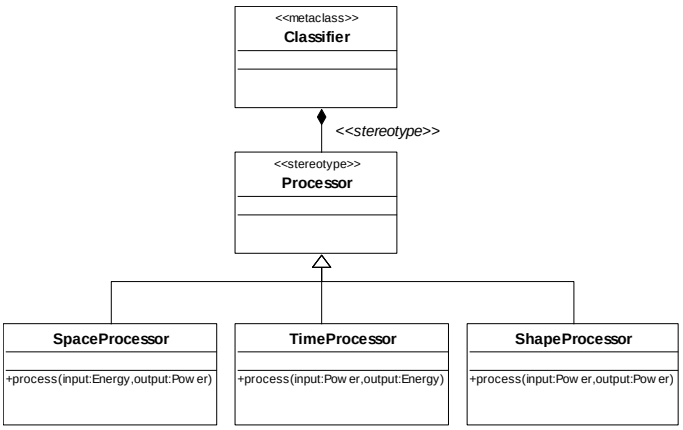

Figure 7. Stereotypes of basic processors

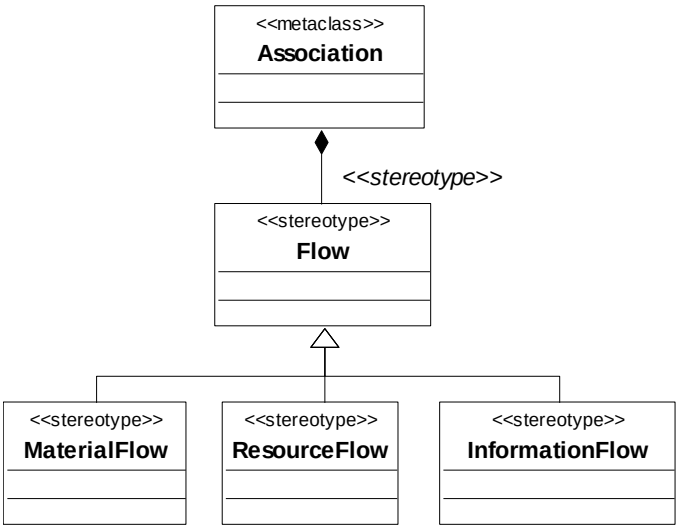

Figure 8. Stereotypes defining the basic flows

As an example of the well-formedness rules applied to each Processor and specially to the TimeProcessor, let have a look to the following one:

*{**context** Processor*

allConnections : Set(AssociationEnd)

let allConnections = self.connection

***inv**: self.allConnections.->select(name='input')->size>=1*

***inv**:self.allConnections.->select(name='output')->size>=1*

}

It states that each Processor shall participate to at least one association which role is named "input". The same constraint applies with a role named "output".

Let's consider the following well-formedness rule:

*{**context** TimeProcessor*

allConnections : Set(AssociationEnd)

let allConnections = self.connection

***inv**: self.allConnections.->select(name='input').*

connection.stereotype->exists(s |

s.name = 'MaterialFlow')

***inv**: self.allConnections.->select(name='output').*

connection.stereotype->exists(s |

s.name = 'MaterialFlow')

***inv**: self.allConnections.->select(name='input').relate.*

oclIsKindOf(PhysicsType)).stereotype.name = 'Power'

***inv**: self.allConnections.->select(name='output').relate.*

oclIsKindOf(PhysicsType)).stereotype.name = 'PotentialEnergy'

}

It states that, if a TimeProcessor participates to an association with another processor, at least one of these associations shall be stereotyped as "MaterialFlow" with a role of "input" or "output" for the considered processor. Moreover, this association shall relate a PhysicsType that is stereotyped as Power when it is an input and PotentialEnergy when it is an output.

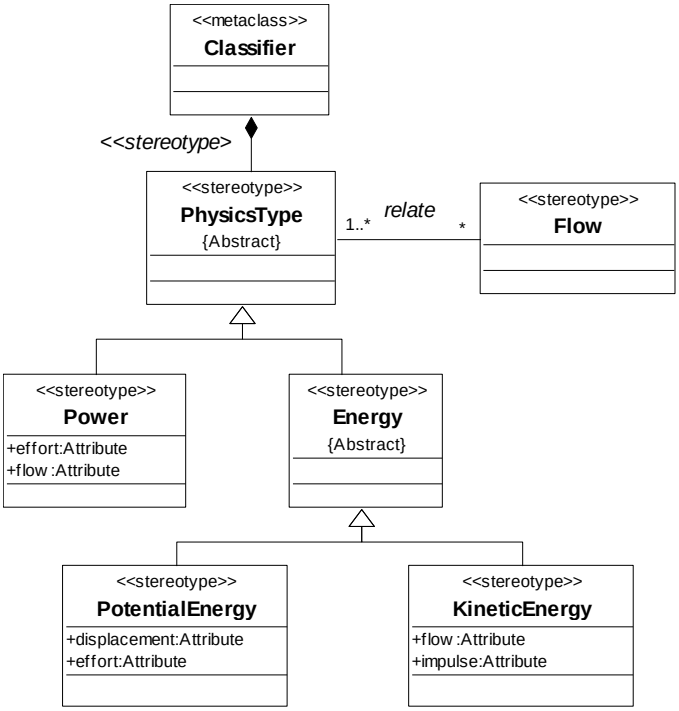

Figure 9. Stereotypes of basic physical variables

The instantiation of these stereotypes, in order to model our case study, is shown in Figure 10. It formalises, and ensures, that, for example, the Storage processor input relates on power defined by the tank pressure as an effort attribute and the tank flow as a flow attribute. Its output relates on a potential energy defined by the tank pressure as an effort attribute and the tank volume as a displacement attribute.

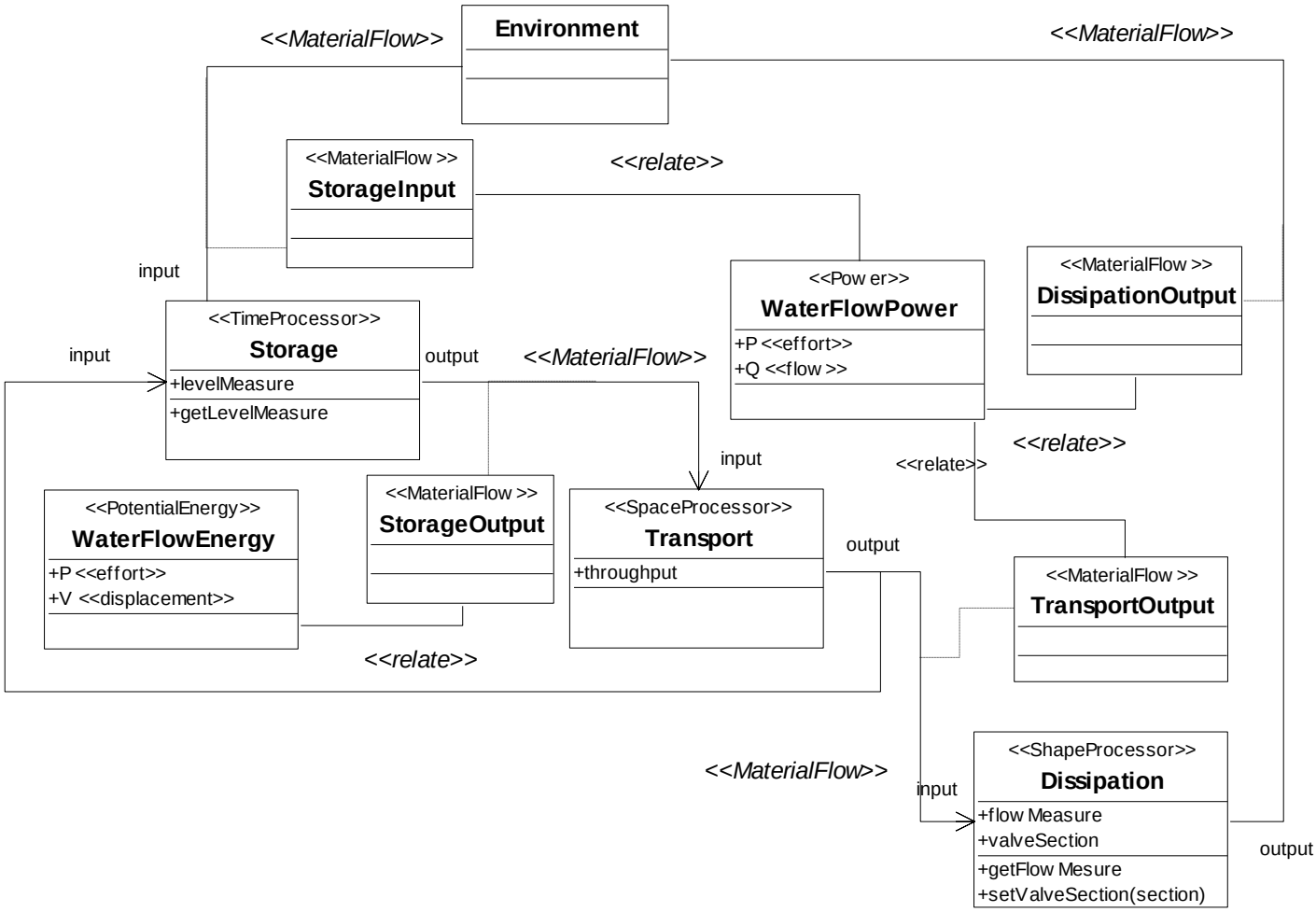

Figure 10. The case study formalisation by instantiation of the basic stereotypes

6 CONCLUSIONS

UML is a general modelling language. Because of this unification, it can be used to model every system, ensuring constraints interoperability. However, it lacks in verification mechanism to ensure the global consistency of the model. Specific profiles, by mean of stereotypes with formalised constraints, help the modeller to build a system model taking into account domain constraints that have to be applied in order to ensure the correctness of the model.

We have proposed, in this paper, a set of UML stereotypes and its constraints that formalise, based on a systemic approach, a framework for the automation of production systems aiming to facilitate a common and consensual understanding of an automated system at the early stage of a systems engineering process [16].

Research work is currently done to translate UML models and OCL constraints into a formal mathematical language in order to verify the consistency of the model.

Future work should be the use of an OCL parser and interpreter, directly into the UML modelling tools, in order to assist the modeller, in real-time, to build a "good" model for his application domain, by mean of the UML profile he will use.

7 REFERENCES

[1] Fusaoka A., Seki H., Takahashi K 1983. A description and reasoning of plant controllers in temporal logic. Proceedings of the International Joint Conference on Artificial Intelligence, pp 405-408. Karlsruhe, 8-12 aug. 1983.

[2] Ramadge P.J. & Wohnham W.M. 1987. Supervisory control of a class of discrete event processes, SIAM Journal on Control and Optimization, Vol. 25 (25), pp 206-230, 1987.

[3] Stiver J., Antsaklis P., Lemmon M., 1996. An invariant based approach to the design of hybrid control systems, Proceedings of the IFAC 13th triennial World Congress., Vol. J, pp 467-472, San Fransisco, CA

[4] Lamboley P., Pétin J.F., Méry D., 1999. Towards a formal engineering framework for process automation, Proceedings of the 7th IEEE International Conf. On Emerging Technologies and Factory Automation (ETFA'99), Barcelone (Spain), 18-22/10/1999, Elsevier Sciences Publisher.

[5] UML 1.4, 2001. Unified Modeling Language, Object Management Group standard, www.uml.org

[6] Panetto H., Pétin J.F., 2003, Setting up UML stereotypes for Production systems modelling. Proceedings of the ISPE-CE2003 conference on Concurrent Engineering : Research and Applications, Madeira, Portugal, 26-30 July 2003, Vol. 1 "Enhanced Interoperable Systems", 747-754, ISBN 90-5809-623-8.

[7] Cassandras C.G., Lafortune S. 1999., Introduction to Discrete Event Systems. Kluwer Academic Publisher, ISBN 0-7923-8609-4.

[8] Lhote F., Chazelet Ph., Dulmet M. , 1999. The extension of principles of cybernetics towards engineering and manufacturing. IFAC Annual Reviews in Control. Volume 23 (1), pp 1-11.

[9] Galara D., Hennebicq J.P., 1999. Process control engineering trends, IFAC Annual Reviews in Control, Vol. 23 (1), pp 1-11, 1999.

[10] MOF Specifications, 1997. Joint Revised Submission, September 1, 1997, OMG Document ad/97-08-14

[11] Panetto H., 2002. UML semantics representation of enterprise modelling constructs, Invited conference, Proceedings of the IFIP ICEIMT international conference, Valencia, Spain, April 24th-26th, 2002

[12] Broenink J.-F, 1999. Object-oriented modeling with bond graphs and Modelica, Proceedings of the International Conference on Bond Graph modeling and simulation, part of the WMC'99, San Francisco, 17-20/01/99.

[13] Isermann R., 1999. Modeling, identification and simulation of mechatronic systems, Proceedings of the 14th World Congress of IFAC, pp 395-496, Beijing, China, July 1999.

[14] Paynter M., 1961. Analysis and design of engineering systems, M.I.T. Press, Cambridge.

[15] Feliot C, Staroswiecki M., 1998. A syntactic approach for functional modelling of physical systems, Proceedings of the 9th International Workshop on Principles of Diagnosis. pp 174-181. Cape Cod (USA), May 24-27 1998

[16] Morel G., Panetto H., Zaremba M.B., Mayer F., 2003. Manufacturing Enterprise Control and Management System Engineering: paradigms and open issues. IFAC Annual Reviews in Control. **27/2**, December 2003.