

FORMALIZATION OF THE DESIGN OF CONTROL SYSTEMS

L. PIETRAC, B. DENIS and J.-J. LESAGE

*LURPA, École Normale Supérieure de Cachan
61, avenue du président Wilson - 94235 Cachan cedex, FRANCE
Tél.: (33-1) 47-40-22-15, FAX: (33-1) 47-40-22-20.*

ABSTRACT

Development of automated manufacturing systems usually consists in varied stages according to different points of view or different subsystems. In each of these stages, different modelling tools are used (often in an integrated manner). Metamodelization give to designers a rigorous way to define modelling tools and their integration. In this paper, we propose a comparative approach between different modelling tools used in the field of metamodelization. In such a context, data modelling tools, algebraic modelling tools, Z language are used to the metamodelization of the Grafcet.

KEYWORDS: models, modelling tool, design integrated methods, metamodelization, Z language, data models, Grafcet

INTRODUCTION

Automated Manufacturing Systems (A.M.S) are complex systems. The performances of A.M.S are closely connected to the quality of the design of their control system. Their design cycle usually consists in varied stages according to different points of view or different subsystems. In each of these stages, different modelling tools are used to design models of control systems, according to specific requirements. In such a context, the coherence between the different designed models is required to guarantee the coherence of the designed control systems.

It is evident that the only textual expression of design methods and of the syntactical expression of modelling tools, is not formal enough to be unambiguous. To get correct and coherent models, we propose to construct models of modelling tools: it is concept of metamodelization. The essential advantages are improvement of reliability, quality, ability evolve, durability, and degree of automation of methods and modelling tools.

In this paper, we are going to present different modelling tools used for the metamodelization in the area of automated manufacturing systems. We underline advantages and disadvantages of existent different approaches. We propose a new approach taking advantage current approach strong points. The formal language Z is to the basis of our approach [11]. To show its interest, we will process a comparative case study on the Grafcet modelling tool.

TECHNIQUES OF METAMODELIZATION

In order that the construction of a model of modelling tools allows their good utilization, the choice of the used modelling tool is essential.

One of the approaches, the most commonly used, is the extended entity/relationship or Niam [3] [6] [4]. That allows the description of concepts used in the modelling tool that is described, as well as the main relationships that implement these concepts. That represents essentially what the modelling tool is able to model.

To improve the description of the organization of a model, object oriented modelling tool have been introduced [1] [8]. Furthermore, that allows a representation of actions that can be process on elements of the model, by means of methods contained in objects (as methods of model construction, or animations of models). However they only specify the existence of an action and its aim, and not its progress. In against part, relationships that link the different concepts of the model are less accurate than entity-relationship or Niam modelling tools can do.

To get detailed actions of animation of a model (as evolution of a dynamic model), some authors use mathematics tools based on algebra [4] [6].

INTEREST OF OUR WORK

The integration and the specification of modelling tools for automated manufacturing system are main aims of this work. To complete this work we have to take an interest in relationships, structure as well as dynamics of modelling tools. We come to see that these three aspects have been processed in current works of metamodelization.

However used modelling tools focus only one some of three aspects. Our approach is based on the utilization of the formal language Z. This language is based on mathematics notions of sets theory, what allows to model concepts and relationships evoked previously. The notion of Z schema allows to structure our metamodels (models of modelling tools). Finally we use operations on set and relationships to model construction or dynamics evolution of models.

CASE STUDY: THE GRAFCET

We are going to illustrate with an example, Z capabilities to make metamodels.

Many works of metamodelization exist on the GRAFCET (Grafcet is a modelling tool used to design the control of logical systems, it's standardized in french and international organizations [5]). Then we have chose it as example, what allow us to compare the different existing approaches of metamodelization.

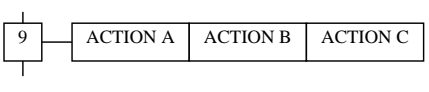
From five extracts of the IEC 848 standard, we present existing metamodels and a Z metamodel. On each case we discuss their respective contributions. These five extracted cases have been chosen to present different metamodelization aspects. These cases are representative of problems met in metamodelization in the field of automated manufacturing systems. Each case is presented according to the following way: extract from the IEC standard (in italic), related metamodels from literature, our Z metamodel. Each case is discuted. The presented metamodels (Z and other) are not extract from global Grafcet metamodel, but are only models from the IEC standard extracts. The coherence between the different metamodels (in a same language) must not be expected.

Actions associated with steps

This part of the IEC standard denotes association between two components of Grafcet. Data models, such as Information Analysis of NIAM [10] or extended entity-relationship, are suitable for modelling this kind of associations. With Niam, fig 1 shows that each step could be

connected to actions, and each action is associated to one or more steps. With extended entity-relationship, on fig 2, authors said that each step could be connected to actions, and each action is associated to one and only one step. Differences between these two metamodells, are not due to a difference of abilities of modelling tools, but are due to a difference of authors' interpretation. This simple example bring to the fore interest of metamodelization.

“... A command (action) is specified by a written or a symbolic statement inside a rectangle connected to the step symbol with which it is associated...”

No.	Symbol	Description
2.4		Three actions A, B and C associated with step 9, horizontal arrangement

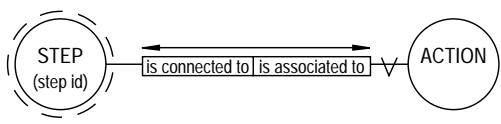


Figure 1. part of metamodel from [7]

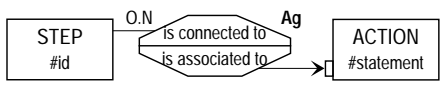


Figure 2. part of metamodel from [2]

$[STEP, ACTION]$
 $lActionStep : STEP \leftrightarrow ACTION$
 $ranActionStep = ACTION$

The Z model we propose has the same semantics as metamodel of fig 3. After the declaration of the two types ($[STEP, ACTION]$) the assertion $lActionStep : STEP \leftrightarrow ACTION$ means that an element of STEP could be in relation with elements of ACTION and vice-versa. Finally, the assertion $ranActionStep = ACTION$ is a restriction of the mathematic relation $lActionStep$ which denotes that range of $lActionStep$ is ACTION: so each element of ACTION is related to one link at less.

Generally speaking, Z could express relations with any kind of restriction (even not binary relations [9]), so it can handle each relation model with Entity-Relationship models.


Active steps and inactive steps

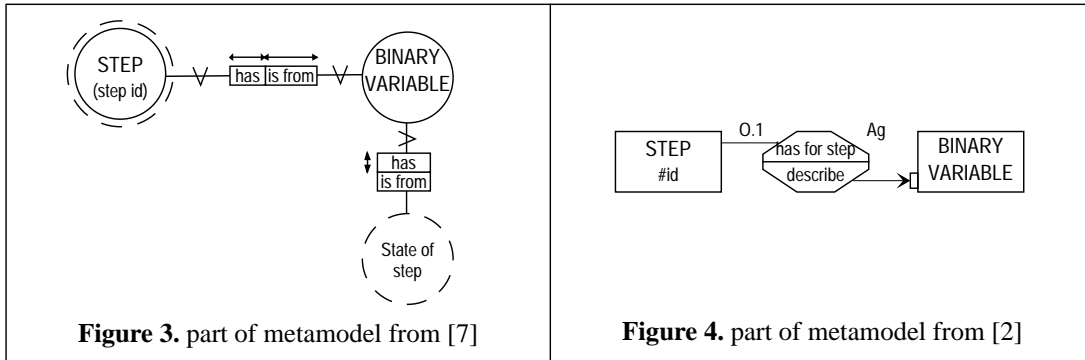
The association STEP-ACTION is a no-temporal relation. The relation between STEP and BINARY VARIABLE is a always true relation, then this relation is correctly expressed in

“... At a given instant a step may be either:

- active, or
- inactive.

The active or inactive state of a step may be represented respectively by the logic values «1» or «0» of a binary variable «X*», in which the asterisk (*) must be replaced by the relevant step label...”

No.	Symbol	Description
1.5		Example: Step 3, depicted in its active state

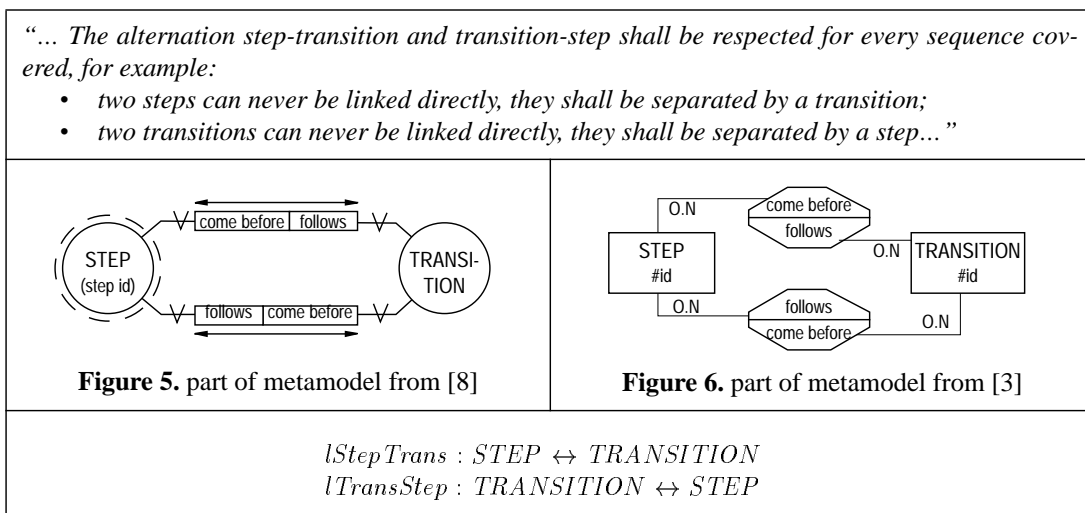


[*TIME*]
activeStep : *TIME* ↔ *STEP*
inactiveStep : *TIME* ↔ *STEP*
 $\forall t : \textit{TIME} \bullet \langle \textit{activeStep}(\{t\}), \textit{inactiveStep}(\{t\}) \rangle$ partition *STEP*

$B == \{n : \mathbb{Z} \mid n = 0 \vee n = 1\}$
BINARYVARIABLE == *TIME* → *B*
lStepVariable : *STEP* → *BINARYVARIABLE*
 $\forall x : \textit{STEP}; t : \textit{TIME} \bullet x \in \textit{activeStep}(\{t\}) \Leftrightarrow \textit{lStepVariable}(x)(t) = 1$

entity-relationship models. On the other hand the state of variable or the state of step are temporal functions. Temporal functions can be represented with **Z** as well as others functions. Furthermore, **Z** model specified binary variables ($B == \{n : \mathbb{Z} \mid n = 0 \vee n = 1\}$) at the opposite data models can only referred it. **Z** model specifies active steps and inactive steps as temporal relations. Ranges of these relations are subsets of **STEP**. Element of **STEP** can only be an element of range of *activeStep* or *inactiveStep*. **Z** language can also express the evolution of variables' state which depend of steps' state.

Syntax rules



There is no problem to represent these relations. But **Z** allows to represent actions on elements in the model. The semantics of the relations "STEP come before TRANSITION" and "TRANSITION come before STEP" is important for the description of the dynamics of the

Grafcet model. The integration of the dynamic in the metamodel allows to accurate the semantic of each relation by the difference of using each relation

The second Z schema present a possible usage of previous relations. This schema represent an operation which specify the construction of a relation between two elements. Elements must be a step or a transition (if not, it is impossible). The selection order is important to make difference between the two relations lStepTransition and lTransitionStep.

$DRAWNOBJECT ::= square(STEP) \mid dash(TRANSITION)$
 $MESSAGE ::= stepToTransition \mid transitionToStep \mid impossible$

<i>LinkAdding</i>	
$firstObject? : DRAWNOBJECT$	
$secondObject? : DRAWNOBJECT$	
$message! : MESSAGE$	
$(firstObject? \in square(STEP) \wedge secondObject? \in dash(TRANSITION) \wedge$ $lStepTransition' = lStepTransition \oplus \{square(firstObject?) \mapsto dash(secondObject?)\} \wedge$ $message! = stepToTransition)$	
∨	
$(firstObject? \in dash(TRANSITION) \wedge secondObject? \in square(STEP) \wedge$ $lTransitionStep' = lTransitionStep \oplus \{square(firstObject?) \mapsto dash(secondObject?)\} \wedge$ $message! = transitionToStep)$	
∨	
$message! = impossible$	

Evolution of active steps

Evolution of active steps can be specify with algebra. But this specification don't use the description of entities. Z allows us to represent this two views of modelling tool. This is a guarantee of the coherence of the metamodel.

The schema "EvolutionOfActiveSteps" presents an operation which determinate set of active steps and actions at time t+1. These sets are determinated from active steps at t and clearing transitions at t+1.

The schema "EvolutionOfActiveSteps" presents an operation which determinate set of active steps at time t+1. This set is determinated from active steps at t and clearing transitions at t+1.

<i>"... The clearing of a transition simultaneously leads to the active state of the immediately following step(s) and to the inactive state of the immediately preceding step(s)</i>	
<p>Evolution:</p> $E_i = A_i + E_i \cdot \bar{D}_i$	<p>With:</p> <ul style="list-style-type: none"> C_j : clearing condition of transition j E_i : state of step i A_i : activation condition of step i D_i : deactivation condition of step i
Figure 7. part of metamodel from [7]	

<p><i>EvolutionOfActiveSteps</i></p> <hr/> <p>ΔInitialSituation</p> <p>$tv : TIME \leftrightarrow TRANSITION$</p> <p>$tf : TIME \leftrightarrow TRANSITION$</p> <p>$above : TIME \leftrightarrow STEP$</p> <p>$below : TIME \leftrightarrow STEP$</p> <hr/> <p>$t' = t + 1$</p> <p>$tv(\{t'\}) = \{x : TRANSITION \mid lStepTrans^{\sim}(\{x\}) \subseteq activeStep(\{t\})\}$</p> <p>$tf(\{t'\}) = \{x : TRANSITION \mid x \in tv(\{t'\}) \wedge transitionCondition(x)(t') = 1\}$</p> <p>$above(\{t'\}) = \{x : STEP \mid lStepTrans(\{x\}) \subseteq tf(\{t'\})\}$</p> <p>$below(\{t'\}) = \{x : STEP \mid lTransStep^{\sim}(\{x\}) \subseteq tf(\{t'\})\}$</p> <p>$activeStep(\{t'\}) = (activeStep(\{t\}) \setminus above(\{t'\})) \cup below(\{t'\})$</p>

CONCLUSION

In this paper, we have underlined the interest of the metamodelization in the area of automated manufacturing systems. The choice of the modelling tools to construct metamodels of a modelling tool is important. We have chosen the formal Z language as modelling tool for the metamodelization. Examples have allowed us to show the aptitude of the Z language for the metamodelization. The Z language has been able to present in a same model the two aspects usually taken into account in existing metamodels i.e. syntax and construction rules matters. Furthermore, Z language is also actually able to handle description of dynamic evolution (always in a same model).

The formal character of the Z language allows to make proofs on models. Our current works tends to use proof capacities of Z to increase the quality of metamodels.

REFERENCES

1. P. Coad and E. Yourdon. *Object-oriented analysis*. Prentice Hall, 1990.
2. F. Couffin and J.-M. Faure. Construction d'un méta-modélisation du Grafset guidée par la structuration. In collective book, Hermès, to appear in 1996.
3. B. Denis, J.-J. Lesage, and G. Timon. Toward a theory of integrated modelling. *Journal of Design Sciences and Technology*, 2(2):87–96, Oct. 1993.
4. J.-M. Faure, P. Lhoste, and J. Z. Jean-Jacques Lesage. Métamodélisation du Grafset. *Automatique Productive Informatique Industrielle*, to appear 1996.
5. IEC. Preparation of function charts for control systems. IEC 848 Standard, 1988.
6. P. Lhoste. *Contribution au génie automatique : concepts, modèles, méthodes et outils*. Habilitation à diriger des recherches, Université de Nancy I, 10 Feb. 1994.
7. P. Lhoste, G. Morel, O. Douchin, and E. Bon-Bierel. Contribution à la méta-modélisation de la syntaxe du modèle Grafset. In collective book, Hermès, to appear in 1996.
8. P. Lhoste, H. Panetto, and M. Roesch. Grafset : de la syntaxe à la sémantique. In *GRAFSET'92*, 13–25, Paris, France, 25-26 Mar. 1992.
9. K. Nguyen and R. Duke. A formal analysis method for conceptual modelling of information systems. In *Int. Conf. on Putting into Practice Methods and Tools for Information System Design*, 93–110, Nantes, France, 10–12 Oct. 1995.
10. G. Nijssen and T. Halpin. *Conceptual Schema and Relational Database Design*. Addison-Wesley, 1989.
11. J. M. Spivey. *Understanding Z: A Specification Language and its Formal Semantics*. Cambridge University Press, 1988.