3 UEML: COHERENT LANGUAGES AND ELEMENTARY CONSTRUCTS DETERMINATION

Matthieu Roque[1], Bruno Vallespir[1], Guy Doumeingts[1, 2]
(1) LAPS/GRAI, UMR CNRS 5131, University Bordeaux 1 - ENSEIRB
351, Cours de la Libération, 33405 Talence FRANCE
matthieu.roque@laps.u-bordeaux1.fr
bruno.vallespir@laps.u-bordeaux1.fr
(2) ADELIOR / Itrec gestion 62bis, Avenue André Morizet
92100Boulogne-BillancourtFrance
doumeingts@itrec.com

Nowadays, one of the important subjects of research in the enterprise modelling domain is the development of a unified language, often called UEML (Unified Enterprise Modelling Language). This paper is focused on one of the more illustrating points about UEML: the comparison of the constructs of the enterprise modelleling language. In previous work we have put in evidence few situations which can occur when we want to compare some modelling constructs belonging to different languages. We investigate more in detail this problem of comparison, in using a formal approach based on the set theory.This paper propses some concepts and guidelines in order to develop UEML.

1. INTRODUCTION

Since the first development in the area of enterprise modelling started in the US in the years of 70's (ex. SADT, SSAD, IDEF0, Data Flow Diagram,…), a lot of enterprise modelling languages have been elaborated world-wide. We can mention for example, Entity Relationship model, MERISE, GRAI grid and nets, CIMOSA constructs and building blocks, OMT, IEM, ARIS method, IDEFx,…(Petit, 1997), (Vallespir, 2003), (Vallespir *et al.*, 2003), (Vernadat, 1996). It is generally recognised that there are too many heterogeneous modelling languages available in the "Market" and it is difficult for business users to understand and choose a suitable one. Main problems related to this situation have already presented in (Chen *et al.*, 2002) and will not explain in this paper. However, it seems that the elements behind these various languages are similar or slightly differ in details. Thus, it is natural to think about the development of a Unified Enterprise Modelling Language. One of the principal benefits to have a Unified Enterprise Modelling Language is to be able to translate a model of an enterprise built in a language in another one (Chen *et al.*, 2002), (Doumeingts *et al.*, 1999), (Vallespir, 2003), (Vallespir *et al.*, 2003), (Vernadat, 2001), (Vernadat, 1999). Moreover, requirements about UEML have been stated during the UEML project (IST-2001-4229) (Knothe, 2003). The third

most important requirement stated was the expectation for an "invariant and unique behavioural semantic" language. Thus, the language UEML is used like a "pivot" language and thus it allows to avoid the one-to-one translation (Chen *et al.*, 2002), (Berio, 2003). Several approaches can be considered for elaborating our unified language like the bottom-up approach which starts with an analysis and then synthesis of existing enterprise modelling languages. Indeed, for the moment, it seems to be more efficient to use the principle which consists in integrating existing languages (Chen *et al.*, 2002), (Vallespir *et al.*, 2003).

In this paper, we only focus on the determination of the common constructs in order to find the elementary constructs. The comparisons of the links between the constructs are not taking into account in these works.

2. DEFINITION OF THE ELEMENTARY CONSTUCTS

In previous works, the concept of elementary construct has been introduced and we highlighted that its determination is not easy (Roque *et al.*, 2005). The objective, of this paper is to propose a formal approach in order to facilitate the determination of the elementary constructs. The definition of the elementary construct is recalled below.

A construct is an elementary construct, if it exists completely or not at all for each considered languages.

For instance, in Figure1, we can see that all the constructs are elementary constructs except the construct C2. This construct belongs completely to the language A but only a part of this construct belongs to the language B. Thus, it is not an elementary construct.

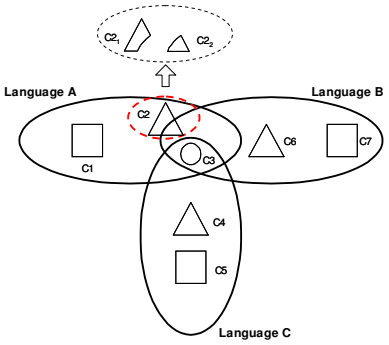

Figure 1 – Elementary constructs

3. CONSTRUCTS COMPARISON

In our approach, we consider the meta-modelling in optics to define a unified enterprise modelling language. Some approaches like XML (DTDs and Schemas), MOF, Telos, can be used as meta-modelling language (Panetto *et al.*, 2004). The meta-modelling language that we use is the UML (Unified Modelling Language) class diagram (OMG, 2003) because it seems sufficient to deal with our problem which is, in first time, to describe the syntactical aspects of the languages. Indeed, for each language, a meta-model[1] is built with the class diagram, in order to represent the constructs of each language. With these meta-models we can compare the constructs of the different languages. Thus, to elaborate the UEML meta-model we have to compare a number N_c of constructs corresponding to all the constructs of the languages. Our objective is to provide a systematic approach in order to determinate which constructs we have to integrate in the UEML language and which are the correspondences rules between them and the constructs of the considered languages. The UEML language is composed by all the elementary constructs which are possible to identify among the N_c conctructs. In order to define these elementary constructs we use an approach based on the set theory approach where each construct is represented by a set.

3.1 Definition of the elementary constructs

Each constructs can be easily represented by a set. Thus, we can write some equations in order to determine the elementary constructs in the case of a number "N_c" of constructs and how the constructs of each language can be recomposed. We can define in the first time the set **E** corresponding to the union of the N_c constructs. Thus, we can define N_{EC} elementary constructs (EC_i) corresponding to all the sub-sets which is possible to create with the intersections of all constructs (1). To determine the elementary constructs, it is useful to use a truth table (as in Boolean algebra) with all constructs. In this table, each "0" corresponds to the complementary[2] of the set in the set E and each "1" corresponds to the set. Thus, each combination of the truth table defines an elementary constructs excepted the first one because ${}^{c}C1 \cap {}^{C}C_2 \cap {}^{c}C_3 = \varnothing$. Thus, in the case of three constructs, we can write the equations below in order to find the elementary constructs and the correspondences rules (see Table 1).

Table 1 – Determination of the elementary constructs

Elementary constructs		Correspondances rules
$CE_1 = C_1 \cap C_2 \cap C_3$	$CE_5 = {}^{c}C_1 \cap C_2 \cap C_3$	$C_1 = CE_1 \cup CE_2 \cup CE_3 \cup CE_4$
$CE_2 = C_1 \cap C_2 \cap {}^{c}C_3$	$CE_6 = {}^{c}C_1 \cap C_2 \cap {}^{c}C_3$	$C_2 = CE_1 \cup CE_2 \cup CE_5 \cup CE_6$
$CE_3 = C_1 \cap {}^{c}C_2 \cap C_3$	$CE_7 = {}^{c}C_1 \cap {}^{c}C_2 \cap C_3$	$C_3 = CE_1 \cup CE_3 \cup CE_5 \cup CE_7$
$CE_4 = C_1 \cap {}^{c}C_2 \cap {}^{c}C_3$		

[1] However, meta-modelling is not an easy step for several reasons: first because given a language it is possible to build different meta-models (as in the case of modelling the same situation) and because there is the need of some guidelines which are not explained in this paper.

[2] equal to $[E - (C_k)]$ noted ${}^{c}(C_k)$

The number of the elementary constructs, in the case of N_c constructs, is given by the equation (1)

$$N_{EC} = 2^{N_C} - 1 \quad (1)$$

3.2 Coherent languages and elementary constructs

The equation (1) does not assume that the intersections between the constructs of a same language are equals to the empty set. Indeed, some languages can have some redundancies or overlapping between their constructs. For the reason, we define the concept of **coherent language**.

A coherent language is a language whose all the intersections between its constructs are equals to the empty set.

Thus, for a coherent language there is no redundancy and no overlapping between its constructs. In the case of the considered languages for elaborating UEML are coherent languages, the number of the elementary constructs can be reduced. Indeed, in this case this number is not equals to (1) but to the equation (2) in removing all the elementary constructs resulting of the comparison of two constructs of same languages.

$$N_{EC} = \left[2^{\sum_{i=1}^{N_L} N_C(L_i)} - 1 \right] - \left[\sum_{i=1}^{N_L} 2^{N_C(L_i)} - 1 \right] \quad (2)$$

Where:

- N_L is the number of the considered languages,
- $N_c(L_i)$ is the number of the constructs of the language L_i.

4. APPROACH FOR DEFINGING THE ELEMENTARY CONSTRUCTS

Finally, we can define three different steps in order to determinate the elementary constructs.

1. Write the equations to define all the elementary constructs for the considered number of constructs.

However, the concept of coherent language of the section 3.2 is very important. In our approach, the definition of UEML is based on the union of constructs of existing languages. For this reason, the problem of redundancy and overlapping constructs of these languages has to be solved before, in order to have simpler and more coherent UEML. In this case, the correspondences rules will be less complicated. Consequently, it seems to be more efficient to apply our approach for defining the

elementary constructs (before the first step), to each language in order to have coherent languages. Moreover, there is no interest to define a unified enterprise modelling language in using languages whose their constructs or part of constructs are not unique in a same language.

2. Interview the providers of the languages in order to identify the intersections between the constructs of the languages.

This step is really not obvious. Indeed, most of the languages have not a formal definition of their constructs. In this case, the comparison is mainly based on informal comparisons where each construct is only defined by a textual description. In the UEML project (Berio, 2003) which provided UEML 1.0, this comparison had been performed by using a scenario. This scenario had been modelled in each considered enterprise modelling language. The study of the intersections between the constructs had been done on the bases of this scenario. Even if, this approach do not provide a formal approach in order to compare the constructs, the lack of formal definition of the constructs, do not permit to use a formal and automatic method. The UEML 2.0 (Berio, 2005) undertakes a very different, eventually complementary approach. Indeed, it requires to fully model the languages in their three conceptual components: abstract syntax, semantic domain and semantics. These three components are organised according to a meta-meta-model: any language is represented by constructs, in turn associated to some meaning provided by a semantic domain. However, the subject of the paper is not to discuss on the way to get the different equations which represent the intersections between the constructs.

3. Resolve the equations according to the results of the preceding step.

5. ILLUSTRATION EXAMPLE

Let us assume that we want to deal with only two pieces of languages: the SADT and the GRAI activities (Roque *et al.*, 2005) as shown in Figure 2.

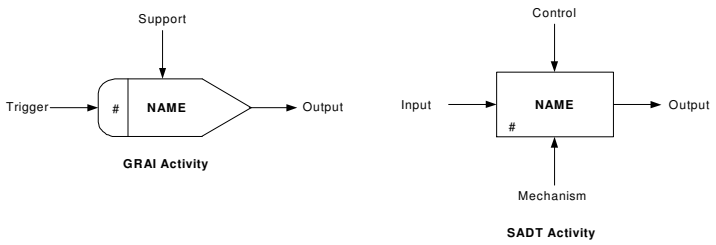

Figure 2 – GRAI and SADT activities

The two simplified meta-models (the links between the constructs of the languages are not represented) of our example are represented in UML class diagrams in the Figure 3. In this paper we focus only on the constructs comparison. In a first comparison, we can identify three elementary constructs which are the Name, the Number and the Output. In the two languages, these concepts are used for

representing the same things. For simplify, these three constructs can be grouped into only one elementary constructs which is called Activity.min (3).

$$\text{Activity.min} = \{\text{Name, Number, Output}\} \tag{3}$$

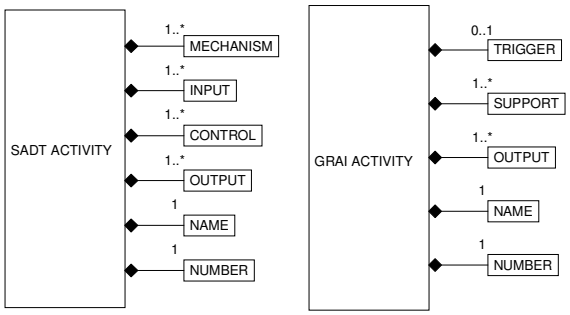

Figure 3 – GRAI and SADT simplified meta-models

5.1 Definition of the elementary constructs and the correspondences rules

5.1.1 First step: Write the equations

Now, we have to consider only five constructs (Support, Trigger, Control, Mechanism and Input) because we have created the Activity.min elementary construct. Thus, with the equation (1) we can define 31 elementary constructs. However, if we use the equations (2) we can reduce this number to 11 elementary constructs. For this example, it is possible to add another assumption in order to reduce again the number of elementary constructs. Indeed, if we take the case of the control, we can see that this constructs is decomposed in three elementary constructs[3]:

$$EC_9 = C \cap {}^{C}M \cap {}^{C}I \cap {}^{C}T \cap {}^{C}S$$
$$EC_{10} = C \cap {}^{C}M \cap {}^{C}I \cap {}^{C}T \cap S$$
$$EC_{11} = C \cap {}^{C}M \cap {}^{C}I \cap T \cap {}^{C}S$$

EC_9 represents a control in SADT which is neither a Trigger nor a Support in SADT. For transformation issue, we can consider that a control can always be linked to a Trigger or a Support. Thus, we can assume that the generalization relationship is complete and that $EC_9 = \varnothing$. We can apply the same principle of all the constructs and finally we have also $EC_1 = \varnothing$, $EC_2 = \varnothing$, $EC_3 = \varnothing$ and $EC_6 = \varnothing$.

$$N_{EC} = \left[2^{\sum_{i=1}^{N_L} N_C(L_i)} - 1 \right] - \left[\sum_{i=1}^{N_L} 2^{N_C(L_i)} - 1 \right] - \left[\sum_{i=1}^{N_L} N_c(L_i) \right] \tag{4}$$

[3] Support → S ; Trigger → T ; Control → C ; Mechanism → M ; Input → I; Not Triggering Control → NTC

The equations (2) can be modified in order to take into account this remark, like is illustrated by the equation (4). With this equation the number of elementary constructs is reduced to 6.

5.1.2 Second step: Interview the providers of the languages

For the five constructs of the two activities, we can write the six relationships below, which will be used to define all the elementary constructs.

1. Trigger $\subset$ Control
2. Trigger $\cap$ Input
3. Trigger $\cap$ Mechanism = $\varnothing$
4. Support $\cap$ Control $\neq \varnothing$
5. Input $\subset$ Support
6. Mechanism $\subset$ Support

5.1.3 Third step: Resolve the equations

In conclusion, we have only 6 elementary constructs. These elementary constructs and the correspondences rules are in Table 2.

Table 2 – Elementary constructs and correspondences rules.

Elementary constructs	Correspondences rules
$EC_4 = {}^{C}C \cap {}^{C}M \cap I \cap {}^{C}T \cap S = I_{UEML}$	$S = I_{UEML} \cup M_{UEML} \cup NTC$
$EC_5 = {}^{C}C \cap {}^{C}M \cap I \cap T \cap {}^{C}S = \varnothing$	$I = I_{UEML}$
$EC_7 = {}^{C}C \cap M \cap {}^{C}I \cap {}^{C}T \cap S = M_{UEML}$	$M = M_{UEML}$
$EC_8 = {}^{C}C \cap M \cap {}^{C}I \cap T \cap {}^{C}S = \varnothing$	$C = NTC \cup T_{UEML}$
$EC_{10} = C \cap {}^{C}M \cap {}^{C}I \cap {}^{C}T \cap S = NTC$	$T = T_{UEML}$
$EC_{11} = C \cap {}^{C}M \cap {}^{C}I \cap T \cap {}^{C}S = T_{UEML}$	

5.2 UEML meta-model and correspondences rules

Finally, we can build the UEML meta-model of this example in UML class diagram (see Figure 4).

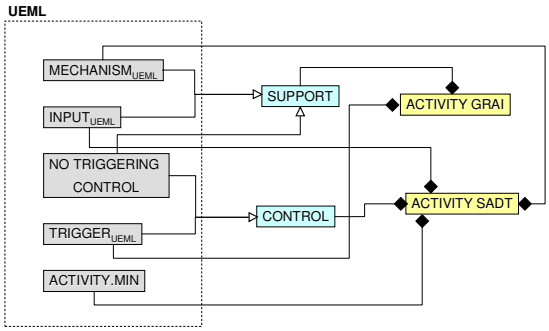

Figure 4 – UEML meta-model and correspondences rules

This class diagram illustrates the UEML meta-model and the correspondences rules between the UEML constructs and the constructs of the GRAI and the SADT activities. Practically, this rule leads to get elementary constructs belonging to UEML that enable to rebuild constructs of languages (so-called local constructs) by generalization. Since these local constructs are obtained, they can be composed to get the whole language.

6. CONCLUSION

In this paper, we have wanted to put in evidence some difficulties concerning the comparison of constructs of enterprise modelling languages. We have presented a systematic approach which provides some help for the determination of the core constructs of the UEML language and the correspondences rules. However, an important question not addressed is the applicability of the methodology for a real case due to the algorithm complexity. Indeed, the number of elementary constructs is of the exponential order and the automatic determination will be difficult without a software support which has to be developed.

7. REFERENCES

1. Berio G., *Requirements analysis: initial core constructs and architecture*, UEML Thematic Network - Contract n°: IST – 2001 – 34229, Work Package 3 Deliverable 3.1, May 2003.
2. Berio G., *UEML 2.0. Deliverable 5.1*. INTEROP project UE-IST-508011 (www.interop-noe.org). 2005
3. Chen D., Vallespir B., Doumeingts G. *Developing an unified enterprise modelling language (UEML) – Roadmap and requirements*. – in Proc. of 3rd IFIP Working conference on infrastructures for virtual enterprise, PROVE, Sesimbra, Portugal, 1st-3 May 2002 – Collaborative Business Ecosystems and Virtual Enterprises, Kluwer Academic Publishers.
4. Doumeingts G., Vallespir B. UEML : *Position du LAP/GRAI. – Seminar of Groupement pour la Recherche en Productique*, GRP, Nancy, France, 25 November 1999.
5. Knotte T., Busselt C. and Böll D. – *Report on UEML (needs and requirements).* - UEML Thematic Network - Contract n°: IST – 2001 – 34229, Work Package 1 Report, April 2003.
6. OMG. *Unified Modeling Language Specification*. – Version 1.5, formal / 03-03-0, 2003.
7. Panetto H., Berio G., Benali K., Boudjlida N., Petit M. (2004). *A Unified Enterprise Modelling Language for enhanced interoperability of Enterprise Models*. Proceedings of the 11th IFAC INCOM Symposium, Bahia, Brazil, April 5-7, 2004.
8. Petit M. *Enterprise Modelling State of the Art*, UEML Thematic Network - Contract n°: IST – 2001 – 34229 – Work Package 1 Report, October 2002.
9. Roque M., Vallespir B. and Doumeingts G. From a models translation case towards identification of some issues about UEML – in Proc. of the workshop on Entreprise Integration, Interoperability and Networking (EI2N), Geneva, Switzerland, February 22, 2005.
10. Vallespir B., Braesch C., Chapurlat V., Crestani D. *L'intégration en modélisation d'entreprise : les chemins d'UEML.* – in Proc. of 4ème conférence francophone de Modélisation et Simulation, Organisation et conduite d'activités dans l'industrie et les services, MOSIM, Toulouse, France, 23-25 April 2003.
11. Vallespir B. - Modélisation d'entreprise et architecture de conduite des systèmes de production. – Thesis for Habilitation à Diriger des Recherches, University Bordeaux 1, 19 December 2003.
12. Vernadat F. UEML: *Towards a Unified Enterprise Modelling Language.* – in Proc. of 3rd Conférence Francophone de Modélisation et Simulation, MOSIM, Troyes, France, 25-27 April 2001.
13. Vernadat F. *Unified Enterprise Modelling Language (UEML).* – IFAC-IFIP Task force Interest group on UEML, Paris, France, 16 December 1999.
14. Vernadat F.B. Enterprise modelling and integration: principles and applications. Chapman & Hall, 1996.