A Model Proposal of the Interoperability Problem.

Vincent Rosener, Yannick Naudet, Thibaud Latour

Centre de Recherche Public Henri Tudor 29, avenue J.F. Kennedy L-1855 Luxembourg
vincent.rosener@tudor.lu, yannick.naudet@tudor.lu, thibaud.latour@tudor.lu

Abstract.. This paper aims at proposing a global view of the interoperability problem, independently of any domain. We first describe explicitly the problem, and the context in which it appears. We then suggest a general structure to handle the possible solutions. Finally, we propose some possible application of our model. **Keywords**: Interoperability, systemic modeling, model, abstraction level, decision-aid.

1 Introduction.

Last year, in the context of the network of excellence INTEROP [1] [2], we have proposed, as a first attempt, a model of the interoperability problem [3]. The main purpose of this work was to suggest a global view of the problem and to identify its key concepts. The present paper shows the latest version of this model. Indeed, we have identified some lacks in the previous model. First of all, it was largely dedicated to software issues. Although it is a very important domain, we thought that enlarging the model to other fields was very interesting in terms of validation. Secondly, the view we adopted about interoperability was too strongly related to communication issues. It did not take into account the structural interoperability.

Starting with these statements, we shall propose an updated definition, as well as associated models and meta-models. All diagrams are written in the Unified Modeling Language [4].

2 The new version of the interoperability model.

We propose in this section an enhanced definition of interoperability, and identify the particular points that necessitate a more formal model. The definition we propose is the following one:

"The interoperability *problem* appears when two or more *heterogeneous resources* are *put together*".

In order to discuss this definition in more detail, we first focus on the important meta-models that will describe the aspects of '*problem'* and '*resource put together'* identified in the definition. As it is a necessary complement to solve interoperability problems, we also propose a meta-model showing the systemic view of modeling.

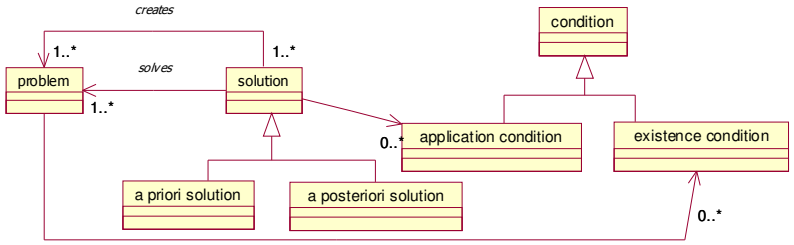

Fig. 1. The decisional meta-model

Interoperability is obviously a *problem*. And there may be some solutions that will solve it. In [3], we proposed a decisional model for problem solving. An enhanced version of this model is illustrated in Fig.1. In this new version, the role of the *condition* concept is strengthened: here, conditions are equally important for the solution and the problem. *Conditions* are essential to take the current context into account. Solutions are therefore dependant on *application conditions* (e.g. the cost), and problems are linked to *existence conditions*. As such, a problem in a given situation might not be a problem in another one. Fig.1 shows as well that there exists *a priori* or *a posteriori* solution with respect to the problem it solves. Thus, a problem can be avoided by anticipation, or corrected after its occurrence.

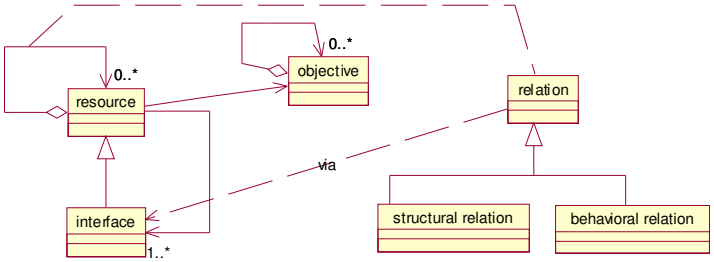

Fig. 2. General overview of resource composition meta-model

Interoperability is about *resources put together*. This part of the definition is illustrated by the resource composition meta-model in Fig.2. More precisely, the interoperability problem appears when putting together resources. By resource we mean tangible things (*e.g.* a piece of steel), or pseudo-tangible things, i.e., that are tangible for a specific environment (e.g. a file is tangible from the OS point of view). The definition of interoperability implies that we consider the relationships that exist between these resources. Fig.2 identifies two kinds of relations: structural relation, which is time independent, and behavioral relation, which is time dependent. For example, a structural relation exists between a plug and an electrical connector. The fact of calling a Web service in an application represents a behavioral relation. The concept of *relation* is a generalization over the *communication* concept highlighted in [3]. In this updated model, communication is a specific behavioral relation.

Fig. 2 introduces two other important concepts: the *objective*, and the *interface* of the resource. The interface is the 'physical' realization of the *relation*. It allows the resource to be used or assembled. Often, and particularly in software engineering, only the interface is considered (port and connector) and seems to be the only focus, as the concern is to know whether tools will be connectible or not. Usually, the objective is not considered. However, the objective really plays a major role. It defines what the resource is useful for. The model in Fig.2 defines an upper abstraction level from which the architectural model of Software Engineering [7] can be derived. Potentially, our proposition could be applied for the building of other type of system.

At this stage, we discussed the fact that interoperability is a problem related to resource in relation. We now need to further discuss the fact that a resource is not only a tangible thing, but also the result of a building process. Indeed, in the context of Engineering, a resource is a result of modeling, and is also part of a global system to build (see [5][6] for a deeper insight on systems and models). The building of a new system leads naturally to complex resource composition [8], which we know being the main cause of interoperability problems.

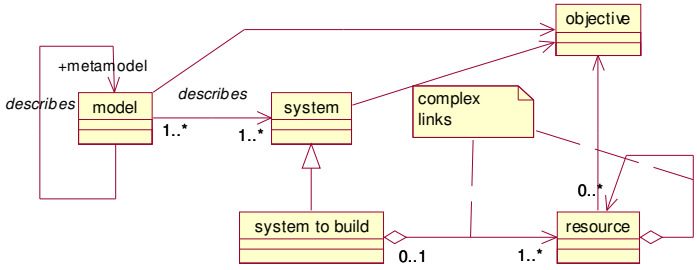

Fig. 3. General overview of systemic meta-model for systems to build

In Fig.3, we show the systemic view that describes the relation between resource, system and model. As stated previously, the introduction of these concepts is very important to handle the 'internal' structure of the resource, in addition to its interface.

The three meta-models we have presented so far are necessary to define the interoperability problem as an instantiation of these models. Fig. 4 illustrates this instantiation and provides a global view of the interoperability problem.

From this view, two different solutions for the interoperability problem are considered [3]: *homogenization*, which can be used when we want to avoid the problem (*a priori* solution), and *bridging* that is used when the problem occurs (*a posteriori* solution). Accordingly to our definition, an interoperability problem appears iff resources in relation are heterogeneous. Therefore, Fig. 4 introduces the existence condition of *resource heterogeneity,* which is considered at the level of the interface or the models of resources.

After having presented the structural aspects of our model, we shall describe its dynamic part. This part will particularly focus on questions like: 'when are the resource heterogeneous?' or ' how to choose between homogenization or bridging?'.

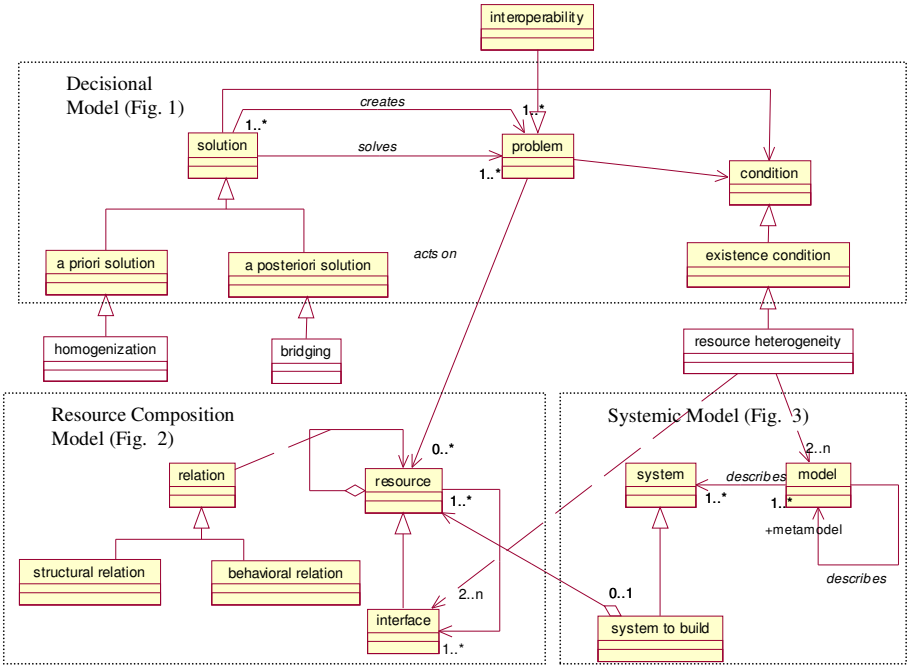

Fig. 4. General overview of the interoperability problem and its solutions

In order to establish the heterogeneity of several resources, it is obviously necessary to compare them. At this stage, it is important to note that there cannot be an interoperability problem at the level of objectives as in Fig. 2. The matching of objectives is another problem pertaining to the engineering (the system building) of the concerned domain. So, even if the *objectives* are key concepts of the resource composition meta-model, they are out of the scope of the Interoperability problem. This one is related to the 'physical' matching of the resources. The check of the interfaces is thus the first step to identify heterogeneity. If the interfaces are not compatible (e.g., the diameter of the screw is different from the internal diameter of the nut, or methods signatures of a library are not compatible with the calling program), then heterogeneity is established and an interoperability problem appears when one decides to relate these resources. This first check should be quite simple as the interfaces are normally clearly identified. The second step consists in the 'internal' compatibility check between the resources. For instance the diameter of screw and nut can perfectly be compatible, but if the screw is made of plastic and the nut of metal, there might be an interoperability problem, as the assemblage could be really defective. To check this level of interoperability, it is necessary to compare the available models of the resources. Having common syntax and metrics will facilitate this operation. If this is not the case, the interoperability model shown in Fig. 4 should

be applied to the resource models themselves. This point will be discussed further in future papers.

After having identified the heterogeneity and the possible interoperability problems that may arise from it, it is necessary to try to solve it. The first important solution criterion is to know if it is possible to modify the resources. If not, then the bridging is the only solution. If the modification is permitted, homogenization is possible. Nevertheless, the modification of a resource is only possible if enough knowledge (in fact, models) is available. If this is not the case, a modeling process should be started. In general, homogenization should be preferred to bridging, as it will ensure a better level of validation of the resulting system with respect to the objectives. The interested reader can find detailed information and examples in [3.]

3 Conclusion and perspectives.

In this brief paper, we proposed a new version of our interoperability model. The meta-models and the considered abstraction level enabled us to work independently of any specific domain.

The work that we currently perform consists in characterizing well-known software architectures against our interoperability model. This task will be particularly useful to help software architects solving classical interoperability issues in relation with other technical problems such as distribution, persistence, or legacy integration.

Besides this top-down validation and use of our model, its instantiation from the INTEROP perspectives (enterprise, architecture, platform and ontological aspects) will also serves as a pragmatic validation. Finally, connecting the models we have presented to some foundational ontologies (such as the ones presented in [9] for instance) would provide an additional bottom-up consistence to our model.

References.

1. INTEROP, European network of excellence (2004). http://interop-noe.org/
2. Interoperability Development for Enterprise Application and Software (IDEAS, European project) (2002). http://www.ideas-roadmap.net/
3. Rosener, V., Latour, T., Dubois, E.: A model-based ontology of the software interoperability problems: preliminary results (2004). In proc. of CAISE04 workshops, EMOI 04, Vol 3 (241-252). http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-125/paper11.pdf
4. The Unified Modeling Language. http://www.uml.org
5. Le Moigne, J.L. : La théorie du système général : théorie de la modélisation. PUF (1977).
6. Minsky, M.L.: Matter, Minds, and Models. MIT Press (1968). http://medg.lcs.mit.edu/people/doyle/gallery/minsky/mmm.html.
7. Garlan, D., Shaw, M.: An Introduction to Software Architecture. Advances in Software Engineering and Knowledge Engineering Volume 2. New York, NY. World Scientific Press (1993) 1-39
8. Simon, H.A.: The science of the Artificial. MIT Press (1969).
9. Sowa, J.F.: Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks-Cole, Pacific Grove, CA, USA (2000)