A MAS Metamodel-Driven Approach to Process Fragments Selection

M. Cossentino[1,2], S. Gaglio[1,3], S. Galland[2], N. Gaud[2], V. Hilaire[2], A. Koukam[2] and V. Seidita[3]

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche, Palermo, Italy
cossentino@pa.icar.cnr.it
[2] Systems and Transport Laboratory (SeT) - Belfort, France
{stephane.galland,nicolas.gaud,vincent.hilaire,abder.koukam}@utbm.fr
[3] DINFO - Università degli studi di Palermo, Palermo, Italy
{gaglio,seidita}@dinfo.unipa.it

Abstract. The construction of ad-hoc design processes is more and more required today. In this paper we present our approach for the construction of a new design process following the Situational Method Engineering paradigm. We mainly focus on the selection and assembly activities on the base of what we consider a key element in agent design processes: the MAS metamodel. The paper presents an algorithm establishing a priority order in the realization (instantiation) of MAS metamodel elements by the fragments that will compose the new process.

1 Introduction

Multi-Agent systems metamodels (MMMs henceafter) and the composition of new design process achieved, in the last years, a greater attention in the agent community. As regards MMMs, the growing importance of Model Driven Engineering approaches required a great effort in the study and modelling of systems on the basis of their metamodels. Besides the effort spent on studying techniques, methods and tools for the production of the right design process meeting specific process requirements (ad-hoc design process for specific situation and development context for solving a specific class of problems), is today more and more increasing. In this field, Situational Method Engineering (SME) [1], provides means for constructing ad-hoc Software Engineering Processes (SEP) by following an approach based on the reuse of portions of existing design processes (often called method fragments[1]). Our work is mainly focused on the use of SME [2–4] for the construction of customized agent-oriented design processes.

In this paper, we show the importance of the MMM in the selection of fragments that will constitute the new SEP, and we explore how MMM could guide in the selection and assembly phases when a new design process is under construction. Selection of fragments is tightly related to the identification of the

[1] From now on in this paper we will use the term *Process Fragment* or simply *Fragment*

new process requirements; fragments included in this process should, in fact, concur to the satisfaction of such requirements. Our thesis is that (some of the) new process requirements are linked to the MAS metamodel elements (MMME) instantiated by the fragments. More details on the structure of a fragment in our approach can be found in [5].

In order to exemplify the fact that many requirements have a natural relationship with some MMMEs, it is sufficient to think about the desired adoption of a goal-oriented analysis rather than an interaction-based one (using use cases). Such different desires will directly bring to the presence of different elements in the MMM (goals rather than use cases). Several other examples can be reported to support this point but we agree that not all requirements can be related to one or more MMME, for instance requirements related to the adoption of a specific practice in performing early requirements analysis will always produce the same output (and therefore will be related to the same MMMEs) but in a different way. In this paper we will only briefly discuss the identification of the MMME starting from the list of the new process requirements. This argument is out of the scope of this paper that it rather focussed on the fragments selection phase and the role of the defined MMM in it. Besides, fragments assembly (that follows fragments selection) is naturally related to the structure of the MMM too, since according to the relationships defined in the MMM, fragments instantiating some elements will naturally need as an input some elements and will produce, as an output, some others that are required elsewhere.

Our proposal consists in the definition of a prioritization algorithm that is used to create an ordered list of MMMEs. This list will be used to select the fragments: the first element of this list will lead to the selection of the first fragment to be imported in the new process and so on for the others (this does not necessarily mean that the fragment related to the first MMME will be positioned at the beginning of the new process life-cycle).

This algorithm establishes an important guideline for method engineers; in other approaches, the selection of fragments is strongly related to method engineer experience [6] or the adoption of complex deontic matrices [7]. The selection of each fragment automatically generates constraints on the following ones, thus giving a great importance to the order used to select them. We are convinced this caused the diffused feeling that method engineering is such a complex discipline that its usefulness is quite limited. With our approach, method engineers do not need a great experience or the capacity to use complex matrices, they only need a well structured repository where MMME can be used for fragments retrieval (we presented such a structure in [8]) and the guidelines we propose in this paper (as already said some activities, namely MMM definition starting from requirements and fragments assembly, are only briefly discussed here, the focus will be on fragments selection).

This article also reports an experiment of creation of a new process (called ASPECS[2]); this is not a classical toy problem but rather we are dealing with the construction of a large process for the design of large multi-agent systems.

[2] ASPECS: Agent-oriented Software Process for Engineering Complex Systems.

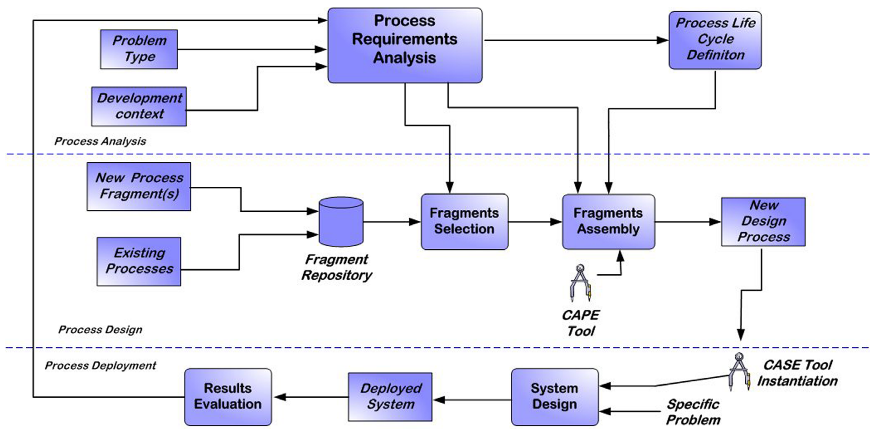

Fig. 1. The PRODE approach for design process composition

The MAS metamodel of this new process [9] is mainly composed by elements coming from the PASSI [10] and CRIO [11] existing design processes and supports Janus as an implementation platforms for holonic agents.

The paper is organized as follows: the next section gives a brief description of the proposed approach. Section 3 lists the requirements from which we started for developing the new process, the experiment done and quickly overview the resulting ASPECS design process. Section 4 discusses similar approaches in the field and, finally, some conclusion statements are provided in section 5.

2 The Proposed Approach

The contribution we propose in this paper is a portion of a complete approach to agent-oriented design process composition. In order to better position this contribution we now describe the overall approach named PRODE (PROcess DEsign for design processes).The PRODE approach is organized in three main phases (see Figure 1): process analysis, process design and process deployment.

Process Analysis deals with requirements elicitation and analysis of the process to be developed. It produces a set of elements, mainly a portion of the MMM, affecting the Process Fragments Selection and Assembly activities. Finally in the Process Deployment phase the new SEP is instantiated, used to solve a problem and then evaluated. Evaluation results are useful for defining new requirements for the next SEP (if any) or improving the designed one. It is worth to note that we consider the process of defining a new design process as an iterative and incremental one.

Process Requirements Analysis is the first activity a method designer undertakes in his work. It has inputs coming from the type of problem to solve. The new process has in fact to be tuned for: *(i)* a specific solution strategy to a class of problem, *(ii)* the development context (composed by the available resources such as people, tools, coding/modelling languages and platforms), and *(iii)* competencies that are available in the SEP enactment group. This activity generates the system metamodel and the other process elements (available stakeholders, required activities/work products) used for the new process creation.

The metamodel contains all the concepts and their relationships. It can be used to design and describe the system under study. It is organized in three different domains, each one being associated to a phase of the development process. The first domain is dedicated to the analysis and provides concepts to describe the problem independently of a given solution. The second provides concepts for the design of a solution independently of a given implementation. And the last one provides platform-specific concepts.

We assume that each concept of the metamodel will be defined (instantiated) in at least one fragment of the process whereas it can be related to other MMME or cited in several fragments. The list of MMMEs is used for the *process fragments selection* from the repository [8][5] as it will be discussed in the next sections. In the *Process Life Cycle Definition* activity, these inputs are also used to define the process life cycle that establishes the structure the designer has to follow during process fragments assembly.

In the **Process Design** phase, process fragments are extracted from existing design processes (or created from scratch) and stored in the Fragment Repository.

In the *Fragments Selection* activity the method engineer adopts the algorithm described in the next subsection for selecting fragments in the prescribed order. The *process fragments assembly* activity results in the new SEP. This activity consists in putting together the selected process fragments according to the structure of the previously identified process life cycle.

This activity is still one of the most important unsolved points in the SME field and some proposal have been done in [12][13]. It is a very complex work where the method designer has to collate all the elements gathered in the previous activities and to merge them by using his experience and skills.

During the *Process Deployment* phase, the system designer adopts the new design process with the help of a CASE tool for solving a specific problem (we developed the Metameth application for such a task). After that the designed system is used and experimented, an evaluation activity occurs in order to evaluate the new design process; gathered information can be used to identify new process requirements for a next iteration.

In section 3 an example on how we apply this process is provided by referring to the construction of the ASPECS process.

In this paper the main focus is on the use of a core part of the MAS metamodel as a guide towards the selection of fragments. The procedure we defined (Figure 2) starts from the identification of the core part of the MAS metamodel that is done by evaluating the contributions that could come from existing design processes or development platforms (in our case they were PASSI, CRIO, JANUS because of our past experiences with them). In fact it is logical to expect that people already skilled with the concepts related to some existing processes or platforms prefers

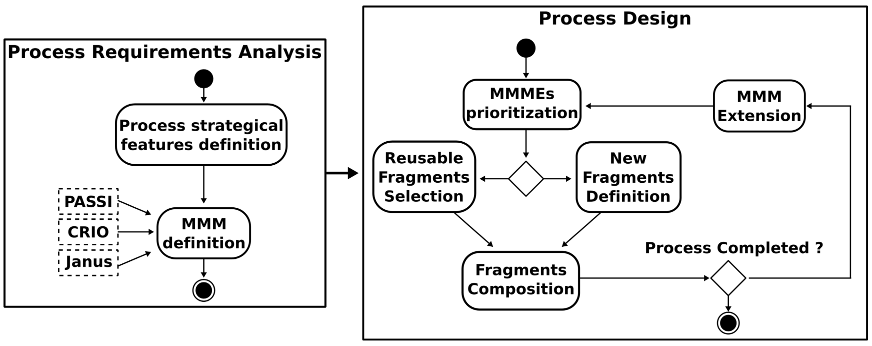

Fig. 2. Details of theprocess requirements analysis and process design phases presented in Figure 1

to reuse them rather than to build everything from scratch. Parts of those metamodels have been reused in order to satisfy the new process requirements that will be described in the experimental part of the paper (section 3).

In the following subsections we discuss the most important steps of this process: the new MAS metamodel construction and the new process design phase where fragments are retrieved from repository and assembled.

2.1 MAS Metamodel Elements Prioritization

As already said, in this work we composed the new metamodel on the basis of portions of metamodels coming from PASSI, CRIO and Janus. In so doing we are aware that defining the core MAS metamodel means defining a relevant part of the 'philosophy' that will be behind the new design process. For this reason we performed this activity during meetings involving stakeholders. We tried to deduct the list of elements by the portions of the cited processes that could satisfy the new process requirements. Of course this was not sufficient and it was therefore necessary to add new concepts for dealing with the specific case. For instance a lot of work has been done in the organizational definition of the agent society holonic structure as well as on the specification of possible roles that could be played by agents inside an holon (Head, Representative, Part/Multipart and StandAlone). These are crucial choices that conditioned the entire process and they have been largely debated before adoption. Some details about the definition of the core metamodel will be provided in the experimental part of the paper in section 3.

The work for designing the new process based on the defined core metamodel can be represented as a cycle composed of three sub-phases as illustrated in Figure 2: *(i)* prioritization of MAS Metamodel Elements (MMMEs); *(ii)* identification and assembly of process fragments defining the MMMEs; *(iii)* extension of the metamodel until the complete process is defined.

The process is detailed in the following algorithm:

```
//Sub-phase 1: MAS metamodel elements prioritization
1. Select a metamodel domain and consider the resulting portion of metamodel as a graph
     with nodes (MMMEs) and links (relationships);
2. Define List_elements as a list whose elements are lists of MMMEs and associated
     priority p: List_elements (p);
       a. p<-1;
       b. List_elements <- null;
3. Produce a linearization of the MMMEs nodes according to a topological sort (elements
     with fewer relationships first) in List_elements, p is the priority index of each
     node in the list
// Sub-phase 2: Selection/Assembly of fragments related to the core MAS metamodel
4. Select/Build fragments for defining (i.e. instantiating) the selected MMMEs by doing:
       a. p<-1;
       b. Selected_el<-List_elements.select(p);
       c. While Selected_el.count>0 do{
           c1.    identify a reusable fragment for the instantiating the first element of
                 Selected_el or create a new one.
           c2. Selected_el.RemoveFirst}
       d. Increment p.
e. Repeat from b.
5. Assembly the fragment in the new process (eventually modify it if required)
6. Select the next metamodel domain (if any) and repeat from 2
//Subphase 3: MAS Metamodel Extension
7. If the process is not completed (i.e. not all design activities from requirements
     elicitation to coding, testing and deployment have been defined)
       a. Introduce new MMMEs
       b. Repeat from 1.
```

The algorithm prescribes (point 3) a linearization of the list of elements of the MMM according to a topological sort criterion. The guideline we propose for accomplishing this task consists in a few steps: consider the elements of the graph, initialize priority p=1, select the element(s) that has fewer relationships with the others, remove it/them (and insert it/them in the List_elements list with priority p), remove the element(s) and all the related relationships from the metamodel, increment p, iterate.

Figure 3 reports an example of application of the proposed algorithm; here a portion of the ASPECS core metamodel is reported, the agency domain. Applying the proposed algorithm we can see that, at the first step, the elements with less relationships are: *Capacity* and *Message*; both these are assigned a level of priority p equal to 1. The next step is to remove these elements from the metamodel (see the right part of Figure 3) and iterate the procedure. The next elements to be considered (p=2) are *Communication* and *Service*, they both have one relationship with *AgentRole*; we can insert them in the List_elements list and remove from the metamodel. Further steps are omitted because of space concerns.

The extension of the core MAS metamodel towards the completion of the process obtained by composing fragments and it should be strongly affected by the awareness of the new process requirements and the relationships among requirements and MMMEs. In extending the initial core metamodel some other criteria should be considered as well. First, the opportunity of reusing existing fragments could lead to the introduction of specific MMMEs related to them.

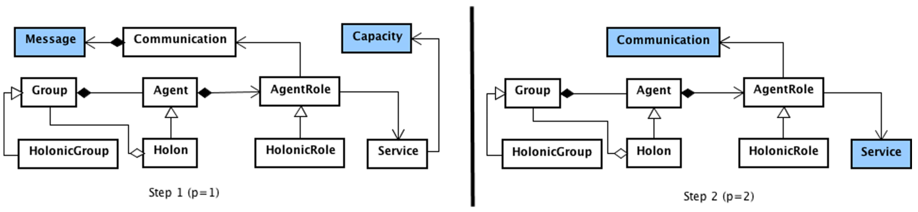

Fig. 3. The first two steps in the prioritization of the Agency Domain elements

This is a kind of bottom-up criterion that privileges the reuse of well-known and tested fragments. Second, as a consequence of adopting a Model Driven Engineering (MDE) approach in the development of ASPECS, we think that: (i) the three identified MAS metamodel domains may be regarded as the three different MDE models; (ii) elements belonging to a domain should have a correspondent element in the following one (correspondence is realized by the transformation from one model to the other). The second rule can have some exceptions related to specific cases when an element is regarded as a design abstraction useful at one specific level but it is not forwarded to the next one. A detailed discussion of criteria and guidelines for MAS metamodel extension is out of the scope of the paper and will be omitted.

3 Building ASPECS

In this section we describe the process we adopted for building ASPECS. We report the process requirements, the initially created core metamodel, the definition of the precedence order of the metamodel elements, the selection/assembly of process fragments and the extension of the metamodel with the consequent selection of new fragments in an iterative process. Finally a short description of the resulting process is provided.

3.1 Requirements for the Construction of ASPECS

The design of the ASPECS methodology has been constrained by a set of requirements that according to the inputs of the process requirements analysis phase presented in Figure 1, can be classified as follows:

(i) *Problem Type*: the scope of the new design process was to develop very large MASs for the solution of complex problems suitable for an hierarchical decomposition.
(ii) *Development context*: the development of the ASPECS methodology can be seen as a joint work of people coming from two different experiences: people working at the SET laboratory who had a strong background in the design and implementation of holonic systems with a strong accent on organizational aspects of MASs (CRIO process) and one new lab member who was the main author of

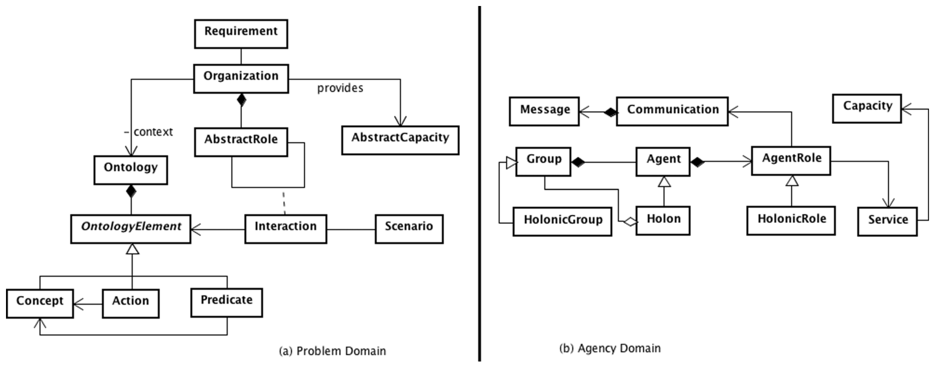

Fig. 4. A part of the ASPECS Problem and Agency domains core metamodel

a process (PASSI, [10]) for the design of MASs where agents were mostly peers and whose important features were: the use of ontologies, a requirements-driven agent identification, the adoption of patterns and tools for supporting design/-coding activities. Participants to this project soon agreed to preserve some key elements of their backgrounds and skills in order to enable an easier transition to the new design process. As regards agents implementation, in the SET lab, the development of a new coding platform Janus was undergoing and its adoption in the new design process was, of course, highly desirable.

These requirements concurred to the definition of the process we describe in the next subsection.

3.2 The Core Metamodel

A part of the initial core metamodel defined for the ASPECS process can be seen in Figures 4(a) and 4(b). This metamodel is a consequence of the process requirements and design choices done during several meetings. Just in order to exemplify how the metamodel was defined we can consider two of the process requirements:

1. An organizational approach was desired were a direct link could be established between the problem and the organization that solves it. This was expected to generate solutions where the (hierarchical) organization structure was an evident decomposition of the problem.
2. A FIPA-compliant communications structure was required.

The first requirement, in the design team opinion, finds a solution in the adoption of a direct link between the Organization and Requirement MMMEs (see Figure 4(a)). This link would represent the direct correspondence between the requirements and the organization that would fulfil them.

The remaining part of the reported Problem Domain core metamodel descends from the definition of organization we decided to adopt: *An organization is defined by a collection of roles that take part in systematic institutionalised patterns of interactions with other roles in a common context. This context consists in shared knowledge and social rules/norms, social feelings, etc and is defined according to an ontology. The aim of an organisation is to fulfil some requirements.*

This definition largely comes from the CRIO approach but it has also been enriched with concepts coming from PASSI (for instance we decided to describe the organizations context by using an ontology). This is coherent with the general requisite of reusing teams members experience as much as possible. The last sentence of the definition is the consequence of the above reported new process requisites.

The second requisite naturally brings to the portion of metamodel reported in Figure 4(b). This was largely inspired by a corresponding portion of the PASSI metamodel where roles, communications, messages are connected in order to satisfy FIPA specifications.

From these and other similar considerations we built the core metamodel for the ASPECS process.

Summarizing, the core metamodel definition process is mainly composed by the following steps: analysis of the process requirements, identification of MMMEs and/or their relationships that could concur to these requirements satisfaction, modification of the core metamodel according to strategic design choice (for instance the adopted definition of Organization reported above).

In the next subsection we discuss the prioritization of the MMMEs representing the order we expect to instantiate these elements in the fragments that will compose the new design process.

3.3 Prioritization of MAS Metamodel Elements

The priority order of the MMMEs was defined by applying the already discussed heuristics to the Problem domain metamodel reported in Figure 4(a). The resulting list is: Requirement, AbstractCapacity, Scenario, AbstractRole, Interaction, Organization, Action, Predicate, Concept, Ontology.

Similarly we obtained a priority order list for the MMMEs elements of the following domains (Agency and Solution). The MAS metamodel of the Agency domain is reported in 4(b).

After this step it is possible to select of fragments from the repository or the construction of new ones in order to define the elements according to the prescribed order. This process will be discussed in the next subsection.

3.4 Selection of Fragments

In performing the fragments selection activity, we refer to our repository of fragments [8]; it includes fragments extracted from PASSI, Agile PASSI, TROPOS, and Adelfe. For the presented experiment we used only fragments coming form PASSI and we purposefully prepared the documentation of fragments coming from CRIO that were not in the repository. Since several MMMEs required by this novel approach (for instance Holon) are not defined by fragments in the repository, we expect to produce several new process fragments, hoping of reusing and modifying some existing ones when possible.

According to the previous discussed list of MMMEs, the first retrieved process fragment is supposed to instantiate the Requirement MMME, a model of system requirements by starting from text usage scenarios. This is exactly what the Domain Requirements Description fragment of PASSI does and it was thus reused. The second MMME (AbstractCapacity) is instantiated by the Capacity Identification fragment reused from CRIO. It is interesting to note that there is no real difference in the precedence order of these two first elements (they share the same value of priority). It is therefore not important to start from one or the other. As already discussed, the two fragments we identified are not necessarily the first two of the process life-cycle. This order in facts arises from the mutual dependencies in terms of input/output among all fragments and could be determined, for instance, by drawing a dependency diagram.

The realization of the third MMME (Scenario) presents an interesting issue: this element is defined in the PASSI Role Identification fragment (where some sequence diagrams are used to describe agent interactions within scenarios). This fragment operates on several different MMMEs: Agent, Role, Actor, Message and its output is the required Scenario. As it can be seen, some of these elements are not part of the core metamodel. This situation (that is quite common) can be solved in two ways: *(i)* the fragment is modified, *(ii)* the elements are added to the metamodel thus enlarging the structure defined in the initial core. Further details on the extension of the core metamodels will be presented in the next subsection.

In a similar way we defined the remaining part of the process. In this discussion we omitted the details of each fragment and the difficulties found in defining the new ones as well as in modifying the reused ones while adapting them to cope with the new specific issues.

In the next subsection we discuss some examples of extension of the initial core MAS meta-model done in order to refine the initial sketch of the process.

3.5 Completion of the Process and Extension of the Core Metamodel

We view the construction of a new design process as an iterative-incremental activity that can be decomposed in the following steps: (i) Construction of a process stub including several fragments. (ii) Test of the process portion. (iii) Evaluation of results. (iv) Next iteration planning in terms of new process requirements to be addressed, changes to be done in the existing process stub, and new parts of the metamodel to be included in the process.

In the ASPECS design process, we performed the first significant test after completing a draft of the System Requirements phase. This test consisted in using the new design process stub for designing a couple of simple applications. As a result of this evaluation, we proposed one change: the explicit introduction of non-functional requirements in the early stages of the process (this implied an extension of the metamodel). After that, according to the 4-steps process discussed at the beginning of this subsection, we designed a new portion of the metamodel, more specifically, the core part of the Agency domain metamodel (see Figure 4(b)). We are not going to detail the work done on this part of the process, we will only discuss one interesting point: the extension of the initially defined core metamodel represented in Figure 4(b) to cope with some new process requirements identified during the iteration. After some evaluations, we realized that in the new process it was not possible to represent not FIPA-compliant agent interactions (for instance environment mediated). They had not been initially listed among the new process requirements but they were already supported by the Janus platform and sometimes used in previous projects developed in the lab. Another issue arose from the consideration that it was not possible to design simple (non holonic) agents like the conventional PASSI ones. This limited the possibility of integrating in the same design, complex holonic hierarchies with simple agents (devoted to deal with minor parts of the problem). In order to solve these issues we changed and extended the core metamodel by including a Conversation and an AtomicAgent MMMEs.

The extended metamodel has been fully realized by a set of fragments and then the process stub tested and evaluated as already described. The work continued in an iterative way until the complete process was defined and thoroughly tested.

Next subsection provides a short description of the resulting ASPECS process[3].

3.6 The Resulting Design Process

ASPECS combines an organizational approach with an holonic perspective. Its target scope can be found in complex systems and especially hierarchical complex systems. The principle of ASPECS consists in analyzing and decomposing the structure of complex systems by means of an hierarchical decomposition. The ASPECS process consists in four phases that are briefly described below.

The *Analysis* phase is based on the identification of a hierarchy of organizations whose global behaviour may represent the system under the chosen perspective. This phase starts with a requirements analysis activity where requirements are identified by using classical techniques such as use cases. Domain knowledge and vocabulary associated to the target application are then collected and explicitly described in the problem ontology. Each requirement is then associated to an organization that represents a global behaviour able to fulfil the associated requirements. The context of each organization is defined by a set of concepts of the problem ontology. The organization identification defines a first hierarchy of organizations that will then be extended and updated during the iterative process. The identified organizations are decomposed into a set of interacting sub-behaviours modelled by roles. The goal of a role is to contribute to the fulfilment of (a part of) the requirements of the organization within which it

[3] A complete description of the ASPECS process can be found at: `http://set.utbm.fr/index.php?pge=352&lang=fr`

is defined. In order to design modular and reusable organization models, roles should be specified without making any assumptions on the architecture of the agent that may play them. To meet this objective, the concept of capacity was introduced. A capacity is an abstract description of a know-how, a competence of an agent or a group of agents. The role requires certain skills to define its behaviour, which are modelled by capacity. The capacity can then be invoked in one of the tasks that comprise the behaviour of the role. In return, an entity that wants to access a role, should provide a concrete realization for each capacity the role requires.

The analysis phase ends with the capacity identification activity that aims at determining if a role requires a capacity. At this step a new iteration may possibly start. If all capacities required by roles at the lowest level of the hierarchy are considered to be manageable by atomic easy-to-implement entities, the process may stop.

The *Agent Society Design* phase aims at designing a society of agents whose global behaviour is able to provide an effective solution to the problem described in the previous phase and to satisfy associated requirements. The objective is, now, to provide a model of the agent society involved in the solution in terms of social interactions and dependencies among entities (holons and/or agents). Previously identified elements such as ontology, roles and interactions, are refined. At the end of the design phase, the hierarchical organization structure is mapped to a holarchy (hierarchy of holons) in charge of its execution. Each of the previously identified organizations is instantiated in terms of groups. Corresponding roles are then associated to holons or agents.

This last activity also aims at describing the various rules that govern the decision-making process enacted inside composed holons as well as the holons' dynamics in the system. All of these elements are finally merged to obtain the complete set of holons (composed or not) involved in the solution. In this way, the complete holarchy of the solution is described.

The *Implementation* phase aims at implementing the agent-oriented solution designed in the previous phase by adapting it to the chosen implementation platform, in our case, Janus. Based on Janus, the implementation phase details activities that allow the description of the solution architecture and the production of associated source code and tests. It also deals with the reuse of previously developed solutions.

The *Deployment* phase is the final one and it aims at detailing how to deploy an application over various Janus kernels. This phase starts with the description of the deployment configuration and details how the previously developed application will be concretely deployed; this includes studying distribution aspects, holons physical location(s) and their relationships with external devices and resources; organization and agent test activities complete the process.

4 Related Works

The work presented in this paper extended the Situational Method Engineering (SME) paradigm until now applied only to the Information System research field. In this section some conventional approaches will be discussed in order to provide an overview of related works.

There are different SME approaches in literature, all of them start by facing the three main phases for the construction of a new design process (process requirement analysis, fragments selection, fragments assembly) and all of them consider the fragment and the repository as fundamental elements. Despite of these works, the problem of how to select and to assemble a set of fragments is still an open issue.

An interesting approach is reported in [14]. Here the method designer firstly identifies the need for the new design process by carrying out a set of analysis activities on the application context. Basing on this analysis the method designer can identify and select a set of fragments that best fit the elicited needs and then he assemblies them. In this process the method designer uses a meta-modelling technique in order to model the design process by using class and data kinds of diagram.

The use of the meta-modelling technique is very similar to what we proposed in this paper, in fact it allows to model all the stages, activities and tasks that have to be performed in the new process thus creating a starting point for the method designer to identify the proper fragments. The main difference with the PRODE approach is that we use the metamodel in order to represent and define a complete knowledge for the new design process, and that its elements are the starting point for the selection from a repository and the assembly of fragments. This technique can be automated and, anyway, it reduces the relevance of designer skills in the process; this kind of dependence is one of the most important problems affecting every approach found in literature.

Another well known approach is proposed by Ralyté et al. [15][3]; the result of the previously said phases is a set of *maps*. These maps can be considered as a guideline during the development of the new design process; they are composed of three elements: source, intention and strategy. The method designer starts from the process requirements specification, through the maps he models the new process at different levels of abstraction and represents the method chunk, in so doing he is able to identify a set of method chunks that satisfy each requirement. Each chunk is equipped with a *descriptor*, done by a set of attributes (ID, name, type, application, etc.); this element lets the designer understand which method chunk can be used in each specific situation.

This kind of approach is quite different from the one we propose because it is more dependent on the method designer skills and knowledge but also it does not provide a knowledge description on the new design process and finally it is necessary to build and use a well defined chunk repository, as descends from the chunk definition and its specific form (the triplet and the descriptor).

Another approach, OPF (OPEN Process Framework) is based on the use of the so called Deontic Matrices [4][16]; after having identified the number and the type of activities to be performed, the workflow among them, the pre- and post-conditions and a first draft of lifecycle, the method designer uses a set of deontic matrices in order to find the relationships among fragments in pair and to be able to select a useful set of fragments. The most important difference and the most important value of this approach is that a very huge repository exists for that and this fact together with the use of deontic matrices allows the designer to cover a well defined path from the first selected fragment to the latest one through the established lifecycle giving, in this way, a real aid to the designer's work. The major problem is the deep knowledge required about the repository that, containing a large amount of fragments, surely cannot be easily acquired nor shared.

The approach presented in this paper has this kind of problem too, but the repository it is related to contains a reduced number of fragments, at an higher level of granularity and with the aid of the illustrated algorithm we claim that it is easier to select the right fragments for assembly.

Finally all the approaches, until now presented, greatly depend on the used definition of fragment our work, instead, aims at providing general methods and techniques that are customizable for every kind of application.

5 Conclusion

Based on the Situational Method Engineering, this paper reports an experiment of creation of a new process called ASPECS. The proposed approach starts from the identification of the new process requirements in terms of development context and problem type. The requirements are used for defining an initial core version of the MAS metamodel. The elements of this metamodel are then ordered in a precedence list, and in this order the fragments are retrieved from the repository and assembled in the new process. The resulting MAS metamodel of ASPECS [9] is mainly composed by elements coming from the PASSI [10] and CRIO [11] existing design processes and supports Janus as an implementation platforms of holonic agents. In previous works applying SME, the method engineer usually selects a set of process fragments that he considers as the best for fitting a particular situation and then modifies or adapts them. The approach described in this paper is different and it aims at being as much free as possible from the designer skills by providing a set of reusable guidelines for fragments selection and assembly.

References

1. Harmsen, A., Ernst, M., Twente, U.: Situational Method Engineering. Moret Ernst & Young Management Consultants (1997)
2. Brinkkemper, S., Welke, R., Lyytinen, K.: Method Engineering: Principles of Method Construction and Tool Support. Springer, Heidelberg (1996)
3. Ralyté, J.: Towards situational methods for information systems development: engineering reusable method chunks. In: Procs.13th Int. Conf. on Information Systems Development. Advances in Theory, Practice and Education, pp. 271–282 (2004)
4. Henderson-Sellers, B.: Method engineering: Theory and practice. In: ISTA, pp. 13–23 (2006)

5. Cossentino, M., Gaglio, S., Garro, A., Seidita, V.: Method fragments for agent design methodologies: from standardisation to research. International Journal of Agent-Oriented Software Engineering (IJAOSE) 1(1), 91–121 (2007)
6. Brinkkemper, S., Lyytinen, K., Welke, R.: Method engineering: Principles of method construction and tool support. International Federational for Information Processing 65, 336 (1996)
7. Firesmith, D., Henderson-Sellers, B.: The OPEN Process Framework: An Introduction. Addison-Wesley, Reading (2002)
8. Seidita, V., Cossentino, M., Gaglio, S.: A repository of fragments for agent systems design. In: Proc. Of the Workshop on Objects and Agents, WOA 2006 (2006)
9. Cossentino, M., Gaud, N., Hilaire, V., Galland, S., Koukam, A.: A holonic metamodel for agent-oriented analysis and design. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 237–246. Springer, Heidelberg (2007)
10. Cossentino, M.: From requirements to code with the PASSI methodology. In: Agent Oriented Methodologies, pp. 79–106. Idea Group Publishing, USA (2005)
11. Hilaire, V., Koukam, A., Gruer, P., Müller, J.: Formal specification and prototyping of multi-agent systems. In: Omicini, A., Tolksdorf, R., Zambonelli, F. (eds.) ESAW 2000. LNCS (LNAI), vol. 1972, p. 114. Springer, Heidelberg (2000)
12. Mirbel, I., Ralyté, J.: Situational method engineering: combining assembly-based and roadmap-driven approaches. Requirements Engineering 11(1), 58–78 (2006)
13. Brinkkemper, S., Saeki, M., Harmsen, F.: Meta-modelling based assembly techniques for situational method engineering. Information Systems 24 (1999)
14. van de Weerd, I., Brinkkemper, S., Souer, J., Versendaal, J.: A situational implementation method for web-based content management system-applications: Method engineering and validation in practice. In: Software Process: Improvement and Practice. John Wiley & Sons, Ltd., Chichester (2006)
15. Ralyté, J., Rolland, C.: An Approach for Method Reengineering. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) ER 2001. LNCS, vol. 2224, p. 471. Springer, Heidelberg (2001)
16. Henderson-Sellers, B.: Process Metamodelling and Process Construction: Examples Using the OPEN Process Framework (OPF). Annals of Software Engineering 14(1), 341–362 (2002)