
The IEEE-FIPA Standard on the Design Process Documentation Template

Massimo Cossentino, Vincent Hilaire, Ambra Molesini,
and Valeria Seidita

Abstract

Nowadays, it is a matter of fact that a “one-size-fit-all” methodology or design process useful and fitting every kind of problem, situation, or design context does not exist. (Situational) Method Engineering (SME) discipline aims at determining techniques and tools for developing ad hoc design methodologies. SME mainly and highly focuses on the reuse of portion of existing design processes or methodologies (*the method fragments*). In order to have means for creating SME techniques and tools and for creating new design processes, some key elements are needed: a unique process metamodel for representing design processes and fragments, a proper template for the description of AO design processes and for the description of method fragments. The FIPA Design Process Documentation and Fragmentation Working Group gave an important contribution to the SME research area in terms of the IEEE-FIPA standard Design Process Documentation Template (DPDT) that provides a standardized template for the description of design processes.

M. Cossentino

Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche, Palermo, Italy
e-mail: cossentino@pa.icar.cnr.it

V. Hilaire

IRTES-SET, UTBM, UPR EA 7274, 90 010 Belfort cedex, France
e-mail: vincent.hilaire@utbm.fr

A. Molesini

DISI - Alma Mater Studiorum – Università di Bologna, Viale Risorgimento 2,
40136 Bologna, Italy
e-mail: ambra.molesini@unibo.it

V. Seidita (✉)

Dipartimento di Ingegneria Chimica, Gestionale, Informatica, Meccanica, University of Palermo,
Palermo, Italy
e-mail: valeria.seidita@unipa.it

1 Introduction

The Design Process Documentation Template (DPDT) is the result of the work done within the FIPA Design Process Documentation and Fragmentation Working Group.¹ This Working Group was created and is inserted in the research context of Method Engineering and Situational Method Engineering for the creation of ad hoc agent-oriented methodologies.

It is currently admitted in both mainstream software engineering and agent-oriented software engineering that there is not a “one-size-fits-all” methodology or design process for designing and developing systems able to solve any kind of problems. It is to date a factual data that software systems are strictly dependent on the environment they will work, on the people who will use them, and above all on the class of problems they will solve; for instance software systems for high-risk military applications are intrinsically different from e-commerce applications, or from health care, etc. The different features that software systems present greatly affect the designer choice about the right design process (or methodology) to use.

Designers often spend a lot of time in studying, adapting, and customizing, when possible, existing design processes, thus also increasing software development costs. It would be useful to have the possibility (techniques, tools, and so on) for the designer to create the design process best fitting his/her needs in a quick and useful way.

In order to overcome this problem, in the 1990s the Method Engineering discipline rose. Method Engineering is the “engineering discipline to design, construct and adapt methods, techniques and tools for the development of systems”; this is the definition generally accepted since 1996 [1]. Then several researchers coined the term Situational Method Engineering (SME) for indicating the part of ME dealing with the creation of method(ologies) (or design process) for specific situations [7, 14, 15]. SME provides means for constructing ad hoc software engineering design processes following an approach based on the reuse of portions of existing design processes, the so-called *method fragments*, stored in a repository called *method base*.

Method fragment is the main concept in an SME approach and several different definitions exist in the literature [1, 5, 9, 13]. All the SME approaches in the literature are based on the assumption that a generic SME process is composed of three main phases: method requirements engineering, method design, and method construction [4]. During the method requirements engineering the method engineer analyzes all the elements useful for identifying the right method fragments to be retrieved from the repository. During method design and method construction phases, the method fragments are, respectively, selected and modified, if necessary, and then assembled in the new method(ology).

One need, raised by this type of approaches, is having fragments description or documentation available in a way that aids a method engineer in easily choosing

¹<http://www.fipa.org/subgroups/DPDF-WG.html>

among a pertinent subset of fragments among existing ones in order to build a new process. To date, several method fragments exist but they are not described in a homogeneous fashion, thus severely affecting the selection and assembly. A unique, versatile, and standard description and documentation of method fragments would inevitably derive from a standard description of design processes.

The FIPA Design Process Documentation and Fragmentation Working Group has three main goals:

- identifying the most suitable process metamodel and notation for representing existing design processes and for representing fragments themselves.
- defining a proper template for the description of agent-oriented design processes.
- defining a proper template for the description of method fragment structure.

An important contribution to the first objective has been identified in the OMG specification, the Software Process Engineering Metamodel 2.0 (SPEM) [11] that meets the definition of design process the WG adopts and provides means for easily representing the elements it is composed of. Roughly speaking, a design process represents *the work* to be done by *roles* in order to deliver *products*. The work to be done is supposed to be divided into three main levels: phases, activities, and tasks. SPEM provides means for representing design processes using the following elements: phase, activity, task, role, and work product. Moreover, in order to have a complete set of concepts for representing design processes and above all because of agent-oriented specific needs, some extensions were needed and were identified in the work of Cossentino and Seidita [2, 12]. Here the notion of MAS metamodel, MAS metamodel concepts, work product kind, and work product content diagram have been introduced.

The IEEE-FIPA standard DPDT [6]—shown below—is the contribution to the second objective of the WG. Indeed, it provides a standardized template that allows the description of design processes. A standardized description is a good feasible way to help method engineers in fragmenting existing design processes and thus choosing pertinent fragments; it is also particularly relevant to researchers and practitioners working on SME approaches. Even if this template addresses the documentation of full processes—it is not intended to document portions of processes—the definition of substantial fragments of a process is facilitated if the whole process has been previously described in a standard way that makes identification and definition of fragments easier. Hence, this phase is preparatory and preliminary to the third WG objective.

The DPDT intends to solve the process description problem by proposing a standard way to describe design processes in order to study and reuse them as they are, or for fragmentation and extraction of method fragments. Even if the idea of this specification was born in the context of multi-agent system design processes, this template has been conceived without considering any particular process or methodology, and this should guarantee that all processes can be documented using the DPDT. Moreover, the DPDT is also neutral in regard to: (1) the MAS (or system) metamodel and/or the modeling notation adopted in the process deliverables as well as in describing the process itself; (2) the content of the process since it may (or not) include activities such as testing, maintenance, and so on; (3) the process life cycle,

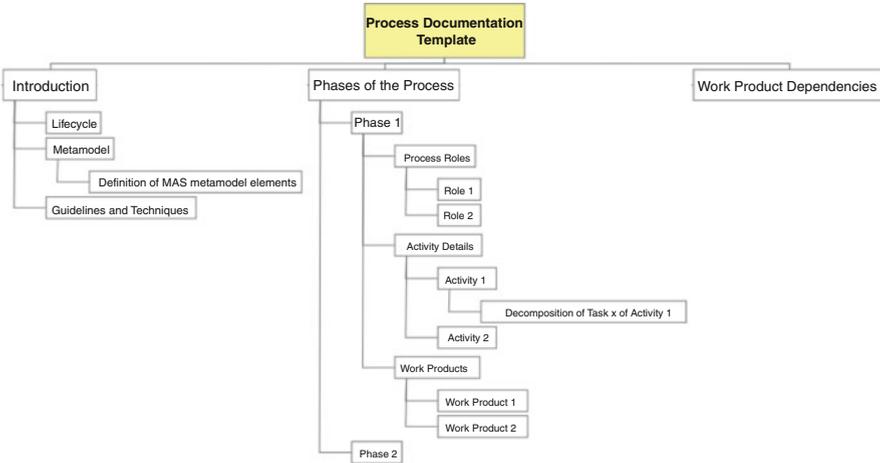


Fig. 1 The design process documentation template structure [6]

since waterfall, iterative-incremental, spiral, and many other kinds of life cycle are fully supported.

The template has a simple structure resembling a tree (see Fig. 1). This implies that the documentation is built in a natural and progressive way, initially addressing the process general description and the metamodel definition, which constitute the root elements of the process itself. After that, the process phases are described as branches of the tree, and also thinner branches like activities or sub-activities are documented. This means the template can support complex processes and very different situations.

Finally, in the DPDT the notation used for modeling some aspects of the process is not considered fundamental. Nevertheless, the use of standards is important. In particular, as already said, the OMG's standard Software Process Engineering Metamodel 2.0 (SPEM) is suggested for modeling such parts of the template. In any case, this does not mean that other standards cannot be used with the template as far as the concepts implied and the underlying view of the system proposed by the work product is reflected in the notation used.

To sum up, the goal of the DPDT is twofold: providing method designers with templates describing design processes in a standard and homogeneous fashion thus highlighting the main concepts they are composed of, supporting the fragmentation process, and the choice of fragments.

In the rest of this chapter we first introduce some basic notions that are useful for understanding the document structure (Sect. 2); then we describe the DPDT outline, and how each part of the document has to be created (Sect. 3).

2 Documentation Template Basic Notions

Before going on with the description of the IEEE FIPA SC00097B specification it is necessary to introduce the reader with some important notions related to the Multi-agent Systems (MAS) metamodel (Sect. 2.1), the set of work product kind used to detail the outcome (work product) of a specific portion of design process, and the work product content diagram that gives the possibility to model all the concepts a specific outcome is devoted to manage/produce (Sect. 2.2).

First of all it is worth noting the notion of design process on which the whole WG's work is based. According to Fuggetta [3], a design process is “*the coherent set of policies, organizational structures, technologies, procedures, and artifacts that are needed to conceive, develop, deploy and maintain (evolve) a software product*”. In such a view, a design process includes components devoted to describe the work—the set of activities and/or tasks—to be done for pursuing an objective, hence delivering some kind of work products, and the stakeholder responsible for doing this work.

The following subsections will introduce the multi-agent system metamodel, some SPEM extensions adopted (better say suggested) to model the process, and the design process documentation template structure suggested by specification SC00097B.

2.1 The Multi-agent System Metamodel

Metamodeling is an essential foundation of Model Driven Engineering (MDE) proposed by OMG² where the language for describing metamodels is the Meta Object Facility (MOF) [8,10] standard. Metamodeling consists in identifying a set of concepts (or elements) to use for describing the properties of models. A model is an instance of a metamodel and it is an abstraction of real world's elements, phenomena, etc. represented in the outcome of the design process phases/activities/tasks. In plain terms, “metamodel is a model of models” that is composed of concepts and relationships among them used for ruling the creation of models.

Some agent-oriented methodologists represent the metamodel of their own methodology by separating the elements into three logical areas: the *Problem domain*, the *Agency domain*, and the *Solution domain*.

The *Problem domain* includes concepts coming from the world where the multi-agent system has to operate; here concepts like requirements, scenario, and so on may be found.

The *Agency domain* collects concepts used to define an agent-based solution for the class of problems the methodology is devoted to address; concepts like agent, role, group, society and communication can be found.

²<http://www.omg.org/index.htm>

The *Solution domain* is composed of concepts representing a well specific mapping among agency domain concepts and the chosen implementation platform, hence the code of the solution.

The metamodel of a methodology gives information about all the concepts that the designer has to manage during his/her work and has to instantiate in, at least one work product. Moreover, the metamodel relationships represent the intra and inter work products dependencies among metamodel concepts, thus establishing logical connections among different parts of a methodology (or of different methodologies).

2.2 The SPEM 2.0 Extensions

The SPEM 2.0 covers almost all the work essentials needed by this working group for representing design processes. Only few extensions to SPEM 2.0 proved to be necessary and were identified in [2, 12]; they extend the elements of the *Process with Method Package* (see [11] for a detailed description of all the SPEM packages). Some details about these extensions will be provided in the following paragraphs.

Work Product Kind

Different processes adopt different modeling styles, thus using different kinds of work products in their flow of work. It is obviously very useful to clearly identify whether a work product follows a specific notation and it points out the structural or the behavioral aspects of the system; or whether it is “intangible” or does not present a specific defined form or more if it aggregates different kinds of work product. What is important to note is that the approach adopted by this working group is hardly grounded on the hypothesis that a work product is something that is produced, consumed or modified during the execution of a portion of work and that each work product serves to model at least one metamodel concept. Moreover, for each metamodel concept drawn in a work product, a set of possible *design actions* may be identified, giving information about the kind of work to be done during a specific portion of a design process.

The set of work product kinds used to detail the outcome of a specific portion of process is:

- *Behavioral*, it is a graphical kind of work product and it is used to represent a dynamic view of the system (for instance an UML sequence diagram representing the flow of messages among agents along time).
- *Structural*, it is a graphical kind of work product too, and it is used for representing a structural view of the system (for instance a UML class diagram).
- *Structured*, it is a text document ruled by a particular template or grammar, for instance a table or a programming code document.
- *Free*, it is a document freely written in natural language.
- *Composite*, this work product can be made by composing the previous work product kinds, for instance a diagram with a portion of text used for its description.



Fig. 2 SPEM 2.0 extensions icons

To complete the presented SPEM 2.0 extensions, the icons shown in Fig. 2 were created.

Work Product Content Diagram

For each work product of the methodology, the work product content diagram gives an overview of the metamodel elements there reported; for each metamodel element the actions to be done are also specified. This diagram is complementary to the textual guidelines useful for drawing the work product.

Design actions that can be made on a metamodel element while composing a work product are:

- *Define*: instantiating an element in the work product. This corresponds to create a new model element. This modeling action is labelled with a **D**.
- *Relate*: instantiating a relationship between two metamodel elements.
- *Quote*: reporting an already defined metamodel element in the work product. Sometimes a newly defined element in a work product has to be related with a previously defined one, this latter element is therefore quoted and related to the defined one. The same action may be done in the metamodel relationships.
- *Refine*: this happens when an already defined element in the work product is in some way refined. For instance, consider to have already defined the A metamodel element, refining that might mean adding an attribute to it in its representation in a class diagram.

Figure 3 shows an example of a work product content diagram; the adopted notation implies a *package* for representing the work product, a *class* for representing the metamodel element, a *label* for representing the action made on the element and *lines* for representing relationships among elements.

The work product content diagrams collect only the elements managed during a portion of design work that are also reported in the work product. During the design process enactment, there can be elements of the metamodel used for reasoning on the output to be produced or used as an input for defining other elements; these elements are not reported in the work product content diagram since they do not appear in the work product.

The DPDT standard does not prescribe the description of one work product content diagram for each activity of the methodology but a unique work product content diagram per phase.

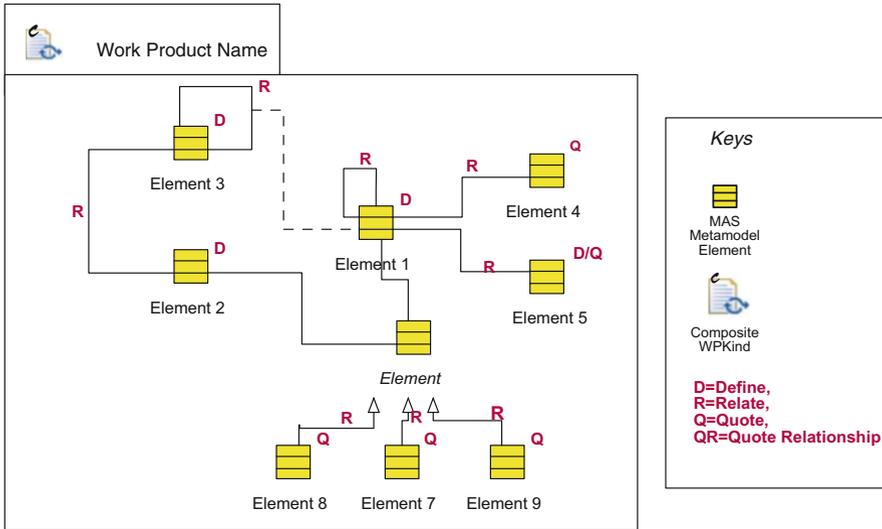


Fig. 3 An example of a work product content diagram

Further refinements and discussions about the notion of metamodel, design actions, and work product content diagram can be found in [2].

3 The Design Process Documentation Template Structure

As it can be seen in Fig. 1 the description of a design process by using this IEEE FIPA standard goes through three main sections: the introduction (see Sect. 3.1), the phases of the process (see Sect. 3.2) and the work products dependency section (see Sect. 3.3). As already said, the work to be done in the process is supposed to be divided into three main levels: phases, activities, and tasks. Phases are composed of activities, which in turn are composed of other activities or tasks. In addition, from a work product point of view, phases are supposed to deliver a major artefact (for instance a requirement analysis document or a detailed design document). Activities are supposed to produce finer grained artifacts (like a diagram often complemented by a text description of the elements reported within it). Tasks are supposed to concur to the definition of activity-level artifacts. The first and the second section of the template contain several subsections, each one structured according to a specific format:

Goal: describing the goal addressed in this part of the documentation. Examples of goals include the documentation of the general philosophy that is behind a design process or the description of the involved stakeholders.

Structure: describing what is to be reported in this part of the process documentation. This may include diagrams as well as the textual description of specific process elements.

Guidelines: describing best practices suggested for a good application of the process documentation template or techniques about how to perform the prescribed work.

Example: addressing an example of outcome produced by the process.

3.1 The Introduction Section

The DPDT *Introduction* section aims at introducing the philosophy, basic ideas, scope, and limits of the process. It is composed of three subsections:

- Global process overview (Life cycle): the aim of this subsection is to organize the process phases according to the selected life cycle (or process model) in order to provide a bird-eye view of the whole process at the highest level of details.
- Metamodel: the aim of this subsection is to provide a complete description of the MAS metamodel adopted in the process with the definition of its composing elements.
- Guidelines and techniques: the goal of this subsection is to provide some guidelines on how to apply the process or to clarify techniques to be used in process enactment whose validity spreads throughout the whole process.

It is very important that the Introduction gives an overview of the methodology from the creators point of view in order to maintain a feeling with the original methodology, thus possibly including original figures, reference materials, and documents.

3.2 The Phases of the Process Section

The *Phases of the Process* section aims at pointing out the process view of the methodology by describing, in a top-down fashion, the activities and comprised tasks each phase is composed of. Moreover, for each phase the list of the involved process roles with the related responsibilities and the list of work products is given.

One different section should be devoted to the discussion of each phase. Indeed, each phase should be studied from a process-oriented point of view: workflow, work products, and process roles are the center of the discussion. The workflow is detailed by means of activities and one subsection for each activity is devoted to describe activities/tasks composing that. The description includes a SPEM activity diagram reporting involved roles (as swim lanes). Further details about each activity can be provided in additional sections. In particular, the subsection discussing each phase should:

- introduce the phase workflow by using a SPEM activity diagram. It reports activities that compose the phase, and it includes a quick description of work products and roles.
- introduce a SPEM diagram reporting the structural decomposition of the activities in terms of the involved elements: tasks, roles, and work products.

The subsection describing each activity should include details about tasks and sub-activities that may be illustrated by a stereotyped UML Activity Diagram. This kind of diagram explains the execution of complicated Tasks by denoting the possible sequences of Steps, which are identified by the «steps» stereotype. Moreover, each diagram is discussed in a separated paragraph that explains the illustrated steps and their relations. Finally a table is required for summarizing:

- Activity name
- Tasks/Sub-activity
- Task/Sub-activity description
- Roles involved

As regards the subsection for the description of work products, it has a twofold aim: (1) detailing the information content of each work product by representing which MAS metamodel elements are reported in it and which design actions are performed on them through one work product content diagram for each phase; (2) describing the notation adopted by the process in the specific work product, also using an example to show it.

Moreover, a table is used to further detail each work product through three columns:

- name of the work product
- description of the content
- classification of the work product according to the already cited paper (categories: Free/Structured Text, Behavioral, Structural, and Composite)

3.3 The Work Products Dependencies Section

The goal of this section is highlighting the dependencies among all the work products produced during the enactment of the design process. Dependencies among work products reflect the dependencies among the portions of work delivering them. Indeed, if the information presented in one work product is used for producing another one, then the two portions of work are obviously related to each other. Work Products Dependencies are represented by a classic diagram (specified by SPEM) called *Work Product Dependencies diagram*.

This diagram can prove to be very important for different reasons, for instance for project managers who have to reschedule project activities according to new needs occurring at design time or for designers who want to keep track of changes in the development process and in the system model documents. For the sake of Situational Method Engineering approaches and for the objectives of this working group the Work Product Dependencies diagram helps the method engineer in identifying dependencies among the fragments that he/she might extract from a design process and above all in identifying the set of MAS metamodel elements each fragment needs.

In fact, it is to be noted that according to the importance that is paid to the MAS metamodel in the Agent-Oriented Software Engineering (AOSE) field, the real input of each portion of a process is a subset of model elements (instances of the MAS metamodel) that constitute the portion of design reported in the input documents.

References

1. Brinkkemper, S.: Method engineering: engineering of information systems development methods and tools. *Inf. Softw. Technol.* **38**(4), 275–280 (1996)
2. Cossentino, M., Seidita, V.: Metamodeling: representing and modeling system knowledge in design processes. In: Proceedings of the 10th European Workshop on Multi-agent Systems, EUMAS 2012, pp. 103–117, 17–19 December 2012
3. Fuggetta, A.: Software process: a roadmap. In: Proceedings of the Conference on the Future of Software Engineering, Limerick, Ireland, 4–11 June 2000, pp. 25–34. ACM Press, New York (2000)
4. Gupta, D., Prakash, N.: Engineering methods from method requirements specifications. *Requir. Eng.* **6**(3), 135–160 (2001)
5. Henderson-Sellers, B.: Method engineering: theory and practice. In: Karagiannis, D., Mayr, H.C. (eds.) *Information Systems Technology and its Applications*. 5th International Conference ISTA'2006, 30–31 May 2006, Klagenfurt. LNI, vol. 84, pp. 13–23. GI (2006)
6. IEEE-FIPA. Design Process Documentation Template. <http://fipa.org/specs/fipa00097/SC00097B.pdf>, January 2012
7. Kumar, K., Welke, R.J.: Methodology engineering: a proposal for situation-specific methodology construction. In: Cotterman, W.W., Senn, J.A. (eds.) *Challenges and Strategies for Research in Systems Development*, pp. 257–269. Wiley, New York (1992)
8. Miller, J., Mukerji, J.: Mda guide version 1.0.1. Technical Report omg/2003-06-01, Object Management Group (2003)
9. Mirbel, I., Ralyté, J.: Situational method engineering: combining assembly-based and roadmap-driven approaches. *Requir. Eng.* **11**(1), 58–78 (2006)
10. MOF. OMG meta object facility home page. <http://www.omg.org/technology/documents/formal/mof.htm>
11. Object Management Group. Software & Systems Process Engineering Meta-model Specification 2.0. <http://www.omg.org/spec/SPEM/2.0/PDF>, April 2008
12. Seidita, V., Cossentino, M., Gaglio, S.: Using and extending the spem specifications to represent agent oriented methodologies. In: Luck, M., Gomez-Sanz, J.J. (eds.) *Agent-Oriented Software Engineering IX*, pp. 46–59. Springer, Berlin (2009)
13. Seidita, V., Cossentino, M., Hilaire, V., Gaud, N., Galland, S., Koukam, A., Gaglio, S.: The metamodel: a starting point for design processes construction. *Int. J. Softw. Eng. Knowl. Eng.* **20**(4), 575–608 (2010)
14. Slooten, K., Brinkkemper, S.: A method engineering approach to information systems development. In: Proceedings of the IFIP WG8.1 Working Conference on Information System Development Process, pp. 167–186. North-Holland Publishing, Como (1993)
15. ter Hofstede, A.H.M., Verhoef, T.F.: On the feasibility of situational method engineering. *Inf. Syst.* **22**(6/7), 401–422 (1997)



<http://www.springer.com/978-3-642-39974-9>

Handbook on Agent-Oriented Design Processes

Cossentino, M.; Hilaire, V.; Molesini, A.; Seidita, V. (Eds.)

2014, VIII, 569 p. 508 illus., Hardcover

ISBN: 978-3-642-39974-9