OSSAP – A Situational Method for Defining Open Source Software Adoption Processes

Lidia López[1(✉)], Dolors Costal[1], Jolita Ralyté[2], Xavier Franch[1], Lucía Méndez[1], and Maria Carmela Annosi[3]

[1] Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
{llopez, dolors, franch}@essi.upc.edu, emendez@lsi.upc.edu
[2] University of Geneva, Institute of Information Services Science, Geneva, Switzerland
jolita.ralyte@unige.ch
[3] Ericsson Telecomunicazioni, Pagani, Italy
mariacarmela.annosi@ericsson.com

Abstract. Organizations are increasingly becoming Open Source Software (OSS) adopters, either as a result of a strategic decision or just as a consequence of technological choices. The strategy followed to adopt OSS shapes organizations' businesses; therefore methods to assess such impact are needed. In this paper, we propose OSSAP, a method for defining **OSS A**doption business **P**rocesses, built using a Situational Method Engineering (SME) approach. We use SME to combine two well-known modelling methods, namely goal-oriented models (using *i**) and business process models (using BPMN), with a pre-existing catalogue of goal-oriented OSS adoption strategy models. First, we define a repository of reusable method chunks, including the guidelines to apply them. Then, we define OSSAP as a composition of those method chunks to help organizations to improve their business processes in order to integrate the best fitting OSS adoption strategy. We illustrate it with an example of application in a telecommunications company.

Keywords: Situational method engineering · Open source software · i-Star

1 Introduction

Open Source Software (OSS) has become a driver for business in various sectors, namely the primary and secondary IT sector. Organizations are increasingly becoming OSS adopters, either as a result of a strategic decision or because it is almost unavoidable nowadays, given the fact that most commercial software also relies at some extent on OSS infrastructure: estimates exist that in 2016, a 95 % of all commercial software packages will include OSS components [1]. OSS adoption impacts far beyond technology, because it requires a change in the organizational culture and reshaping IT decision-makers mindset. Hence, the way in which organizations adopt OSS shapes their business processes. In this context, methods for defining business processes that tailor organizations to OSS adoption consequences are needed.

S. Nurcan et al. (Eds.): CAiSE 2016, LNCS 9694, pp. 524–539, 2016.
DOI: 10.1007/978-3-319-39696-5_32

In this paper, we propose OSSAP, a method for defining **OSS** **A**doption business **P**rocesses. The objective of OSSAP is to model the business processes that an organization needs in order to adopt OSS according to its strategic needs. In order to consider the variability of these strategic needs and the multiplicity of organizational situations to be taken into account, we use Situational Method Engineering (SME) [2] as approach to design our method as a composition of method chunks. In particular, we use the assembly-based SME approach that allows us to combine two well-known modelling frameworks, namely goal-oriented models (using *i** [3]) and business process models (using BPMN [4]) together with guidelines that focus on the OSS adoption strategies and its business processes. As a preliminary step, we will identify and define a set of method chunks to be used in this assembly-based approach.

The rest of the paper is organized as follows. Section 2 provides the background and general methodology of the paper. Section 3 describes the creation of the method chunks needed in our approach while Sect. 4 presents the design of the complete OSSAP method. Section 5 details an example of the application of the new method. Finally, Sects. 6 and 7 present discussion, conclusions and future work.

2 Background on SME and OSS Adoption

2.1 Situational Method Engineering

The discipline of Situational Method Engineering (SME) [2] promotes modularization and formalization of method knowledge in the form of autonomous and interoperable method components, and their composition into new methods taking into account the specific situation of the organization/project at hand. Such a modular definition of methods allows to achieve a better flexibility in method application and to ensure that the method takes all engineering situations into account and provides the best fitting guidance for each of them.

A detailed state of the art of the SME domain reveals various formalisations of method components as well as their assembly techniques [6]. For constructing the OSSAP method we apply the *assembly-based SME approach* [7] that supports new method construction as well as method extension by applying three steps: method requirements specification, method chunks selection and assembly of the selected chunks. Method chunks are reusable method components. A method chunk combines method process (i.e., the guidelines provided by the method chunk) and its related product knowledge (i.e., the formalisation of concepts and artefacts used by the method chunk). A method chunk also includes the situation in which it can be applied (i.e., the required input artefacts) and the intention (i.e. the engineering goal) to be reached.

Method chunks can be identified and defined in different ways. For instance, they can be created by reengineering existing methods into sets of reusable method chunks organized as strategic process models [8]. This reengineering variant (hereafter *reengineering SME*) is founded on the Map process modelling formalism [9], which allows to express methods in terms of intentions, and strategies to reach the intentions, instead of fixed steps and activities. Since many strategies can be defined for achieving an intention, Map allows to represent complex, flexible and situation-driven process

models including multiple ways to achieve method intentions. Every section (i.e. a triplet <source intention, strategy, target intention>) in the process map is then assessed whether it represents autonomous and reusable method knowledge and in this case it is formalised as method chunk. If some map sections are not considered as such, the method map should be refined (e.g. by merging some intentions). Identified method chunks can be atomic or aggregate.

When no method exists, the *ad-hoc SME approach* [10] is more appropriate. In the ad-hoc approach, a method chunk is discovered as a means to satisfy some specific modelling purpose: the specific modelling domain must be analysed and method requirements supporting the engineering of this domain must be identified.

2.2 OSS Adoption

OSSAP builds upon a previous work [5] that we name *the DKE-approach* (after its publication venue) where we proposed a catalogue of *i** models to represent different OSS adoption strategies. These strategies were formulated by assigning in different ways the concepts of an OSS ontology into two actors that belong to an OSS ecosystem: the adopter organization and the OSS community that delivers the software.

The catalogue of adoption strategies is described in [5]. In short: (1) *OSS acquisition* consists in using existing OSS code without contributing to the underlying OSS project/community; (2) *OSS integration* implies the active participation of an organization in an OSS community with the purpose to share and co-create OSS in order to benefit from the commonly created OSS components; (3) *OSS initiative* consists in initiating an OSS project and establishing a community around it over which control is exercised; (4) *OSS takeover* means to take over an existing OSS project/community and to control it; (5) *OSS fork* consists in creating an own independent version of the software that is available from an existing OSS project or community; (6) *OSS release* implies that the organization releases bespoke software as OSS but does not care whether an OSS community forms around it.

2.3 Overall Strategy for Designing the OSSAP Method

As commented above, we will use the assembly-based SME approach to deliver the OSSAP method; this will be explained in detail in Sect. 4. Since the second step of assembly-based SME requires the selection of existing method chunks, in Sect. 3 we will construct such a catalogue in the basis of the needs of our method: some chunks for OSS adoption and some for process models:

- For the first subset, we will apply reengineering SME to the DKE-approach. The resulting subset supports the business analysts during the process of obtaining an *i** model for OSS adoption tailored to the strategic needs of a specific organization.
- For the second subset we will apply the ad-hoc SME approach. These new method chunks guide the analysts to obtain the BPMN business processes that implement the strategic goals from such *i** model.

3 A Catalogue of Method Chunks for the OSSAP Method

In this section we describe the creation of the method chunks that are used as starting point to design the OSSAP method that will be presented in Sect. 4. First, we focus on the method chunks for obtaining the OSS adoption strategies and then on those for obtaining the OSS business processes to implement them.

3.1 Method Chunks for Defining OSS Adoption Strategies

We have applied the re-engineering SME method [8] on the DKE-approach [5]. As explained in Sect. 2.1, the reengineering SME recommends to redefine first the process model of the existing method by using the Map formalism. Then the process map sections are formalised as method chunks. We develop next these two steps.

Step 1: OSS Adoption Process Map Construction. The DKE-approach is described in detail in [5]. Its process map is shown in Fig. 1. The initial intention is to document the organization business and its strategic goals (I1). As suggested by the DKE-approach, this intention is achieved by using *i** goal-oriented modelling as strategy (S1). Then, the DKE-approach proposes a two-step process with intentions: selecting the appropriate OSS adoption strategy from a predefined set of candidates (I2) and upgrading the organizational goal model with the goals defined in the selected strategy model (I3). To satisfy intention I2, the DKE-approach proposes (S2) the catalogue of OSS adoption *i** models described in Sect. 2.2 and a set of coverage metrics that measure the similarity of each of them with the organizational model. I3 is achieved by merging the organizational goal model with that of the selected strategy (S3).

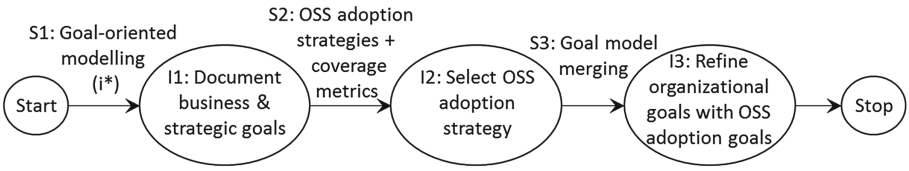


Fig. 1. Process map for defining OSS adoption strategies

Step 2: Method Chunks Identification and Construction. As explained in Sect. 2.1, the identification of method chunks is based on the analysis of the process map sections. The process map resulting from Step 1 is composed of three sections: <Start, S1, I1>, <I1, S2, I2>, <I2, S3, I3>. We consider each of these map sections as reusable method knowledge and accordingly we identify three method chunks:

- MC1: Goal modelling with *i**. It corresponds to the *i** modelling framework [3].
- MC2: OSS adoption strategy selection. Provides the guidelines to select OSS adoption strategies as described in the DKE-approach [5]. Since these guidelines include a catalogue of candidate models (see Sect. 2.2), MC2 can be considered as

an aggregate method chunk and each of the six OSS adoption strategy models as a sub-chunk, MC2.1-MC2.6 (e.g., MC2.1 is OSS Acquisition adoption strategy).
- MC3: *i** model merging. Provides the guidelines to merge goal models as described in the DKE-approach [5].

We present three of the identified method chunks using a tabular representation based on the method chunk metamodel [8]. The process and product parts are presented in an abridged form. Table 1 describes the MC2 aggregate method chunk. Its process part guides the business analyst to select the adoption strategy which best covers the organizational goals. For each adoption strategy there is a corresponding sub-chunk.

Table 1. Method chunk for selecting an OSS adoption strategy

Identifier	MC2: OSS adoption strategy selection
Situation	Goal model representing organizational goals using the *i** framework
Intention	Select an OSS adoption strategy by using coverage metrics
Process part 1. Evaluate the coverage metrics using the organizational goal model and each of the following method chunks: "OSS Acquisition adoption strategy", "OSS Integration adoption strategy", "OSS Initiative adoption strategy", "OSS Takeover adoption strategy", "OSS Fork adoption strategy" and "OSS Release adoption strategy". 2. Select the most suitable adoption strategy according to the resulting measures. Some qualitative evaluation among similar coverage results can be needed.	***Product part*** • *i** model corresponding to the selected OSS adoption strategy (defined in the corresponding sub-chunk). • Definition of the coverage metrics provided by the DKE-approach (see [5], Sect. 6.1).

Table 2 presents the sub-chunk MC2.1 for the OSS Acquisition adoption strategy. All the sub-chunks for adoption strategies share a similar structure: situation: represents the decision to adopt the strategy; intention: documenting the goals related to the strategy; process: application of the proposed model; product: the *i** model representing the strategy. The Acquisition strategy implies to use OSS without contributing to the supporting OSS community. The product model shows how the OSS adopter only obtains and uses the component from the OSS community and does not give back any return to it. Therefore, only outgoing dependencies stem from the adopter actor and it depends on the community to obtain the OSS component and its documentation.

Finally, Table 3 describes the method chunk to refine organizational goals with the goals of an adoption strategy (MC3). Its process part consists in the application of guidelines to merge goal models. This method chunk is described in a way that can be applied to any context that requires merging two *i** models, making it highly reusable.

3.2 Method Chunks for Defining OSS Business Processes

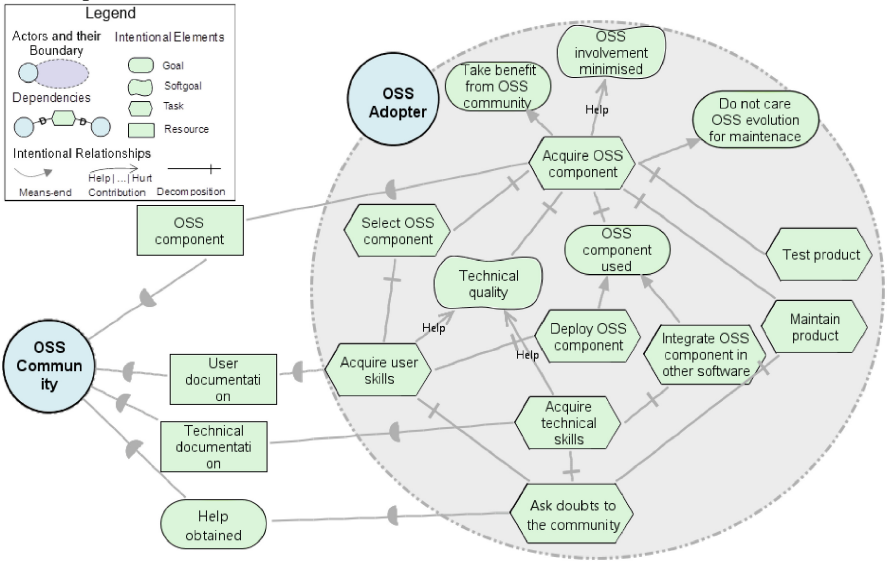
In this section we describe the creation of method chunks for obtaining the OSS business processes that an organization should implement to attain the goals of its selected OSS adoption strategy. To our knowledge there are not existing proposals to define business process models for OSS adoption, so the reengineering SME method applied in Sect. 3.1 is not applicable. Instead, we have applied the ad-hoc approach in which the method chunk construction is made from scratch (see Sect. 2.1) [10].

Table 2. Method chunk for the OSS Acquisition adoption strategy

Identifier	MC2.1: OSS Acquisition adoption strategy
Situation	Decision to acquire OSS
Intention	Documenting goals related to the OSS Acquisition adoption strategy

Process part: use the proposed goal model for documenting organization's goals.

Product part:



Method Chunk Identification. We have elicited the goals that represent requirements that the adopter organization must fulfil to apply each strategy from the adoption strategy *i** models (one shown in Table 2 and the rest available in [5]). These goals have led to the identification of method chunks for defining a specific OSS business process aimed at their satisfaction. In Table 4, we list those goals as method requirements together with their associated method chunks. For instance, the goal *OSS component used* from the OSS Acquisition strategy (see Table 2) has yield to the requirement *Defining business processes for using an OSS component* (third row in

Table 3. Method chunk for merging two *i** goal models

Identifier	MC3: *i** model merging
Situation	Two goal models which are conceptually overlapping
Intention	Merge two related goal models into a more general one, by unifying intentional elements that are shared in both of them
Process part 1. Merge both models applying a semantic similarity notion (see the DKE-approach as example [5]). 2. Making the necessary adjustments to the resulting model in order to resolve any possible inconsistency or ambiguity.	***Product part*** • Two *i** models with some conceptual overlap. • Definition of the merge rules provided by the DKE-approach (see [5], Sect. 6.2).

Table 4). There are three method chunks for it because the adoption strategy *i** models [5] include different ways to achieve it, depending on whether the component is simply deployed or it is integrated as part of another software artefact or, in the latter case, depending on whether the component is redistributed or not (i.e., OSS licenses define different rights for the case of redistributing the software [11] since the distributed software needs a license compatible with the OSS component license and the licenses of the OSS components inside it). The last row of the table provides the requirement

Table 4. Method chunks for Defining OSS business processes

Method requirement	Method chunk identified
Defining business processes for developing a new OSS component	MC4: Creating OSS
Defining business processes for selecting an OSS component	MC5: Selecting OSS
Defining business processes for using an OSS component	MC6: Deploying OSS
	MC7: Integrating and redistributing OSS
	MC8: Integrating OSS without redistributing it
Defining business processes for contributing to an OSS community	MC9: Reporting bugs about OSS
	MC10: Patching OSS
	MC11: Supporting OSS Community
Defining business processes for exercising the leadership of an OSS community	MC12: Leading OSS Community
Defining business processes for creating a community around an OSS component	MC13: Creating OSS Community
Defining business processes for OSS adoption	MC14: Defining OSS Adoption Business Processes

Defining business processes for OSS adoption which embraces all the previous ones and leads to the identification of a method chunk which is the aggregation of all the rest which can be seen as its sub-chunks.

Method Chunk Construction. When constructing new method chunks from scratch, theory plus best practice facilitates the initial definition of chunks [6]. Therefore, we have based our method chunk construction on the allocation of OSS adoption activities and resources from the OSS RISCOSS ontology [5, 12] (partially based on OFLOSSC [13]) to the method chunks; in other words, the business processes related to the method chunks should include the allocated activities and resources. The allocation has been based on the RISCOSS ontology definitions together with the expert knowledge from the RISCOSS EU-funded project industrial partners (www.riscoss.eu). Table 5 provides this allocation for one of the method chunks that we have identified, namely *MC10: Patching OSS*. According to the ontology, patching OSS refers to the development of a patch to correct some bug or add some new feature to an OSS component.

Table 6 describes the *Patching OSS* method chunk. Its situation reflects that it must be applied when an organization has as part of its adoption strategy the goal of providing patches to an OSS community. Its product part consists in a BPMN diagram with the activities and resources allocated to the chunk organized in a process.

Table 5. Allocation of activities and resources from the RISCOSS ontology

Method chunk	Activities	Resources
Patching OSS	Develop Patch, Test, Discuss Solutions, Commit Code, Send Patches, Acquire Legal Skills, Acquire Technical Skills, Acquire Management Skills	Patch, Solution Message, OSS License, Administrator Manual, API Documentation, Defect List, Developer Manual, Release Note, User Manual, Governance Documentation

This chunk has activities devoted to acquire the needed skills to develop patches for OSS, activities needed to develop the patch and reporting it to the OSS community and, in case the adopter organization is allowed to do it, the commit to incorporate the patch to the OSS component. All these activities come from the RISCOSS ontology except for: (1) *Acquire Community Practice Skills* and *Acquire Technical Quality Knowledge* which, actually, specialise an activity from the ontology, *Acquire Management Skills*, because only the part of the governance documentation related to community practices and quality policies is needed by the adopter to know how to develop the patching process and (2) *Report Patches* which is a specialization of *Discuss solutions*. All resources come from the RISCOSS ontology although the resource *Governance documentation* has been split into three: *Licensing Policies, Quality Policies and Community Practices* in order to distinguish the different parts of the governance documentation that are needed for different activities.

Table 6. Description of the patching OSS method chunk

Identifier	MC10: Patching OSS
Situation	Patching OSS is an organizational goal
Intention	Defining OSS adoption business processes by contributing to an OSS community

Process part: use the proposed process model.

Product part:

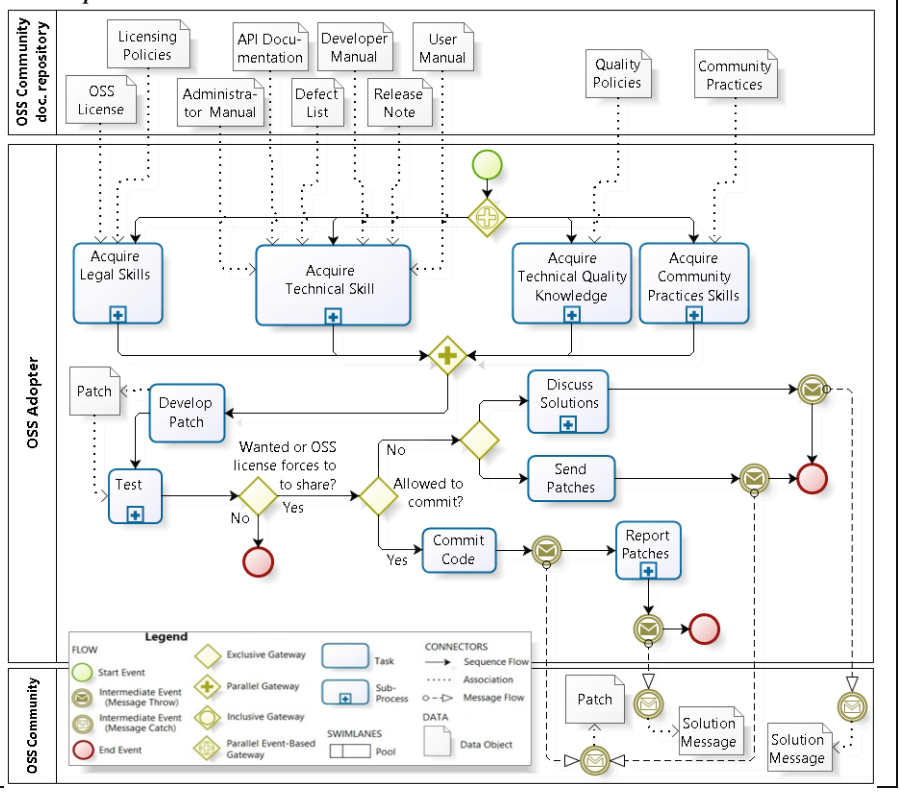


Table 7 describes the aggregate method chunk *MC14: Defining OSS Adoption Business Processes*. Its process part provides the criteria to discriminate which of the chunks for defining OSS business processes (MC4 – MC13) must be applied in a specific case according to the strategic goals of an organization.

4 OSSAP Method Design

To design the OSSAP method we apply the assembly-based approach outlined in Sect. 2.1 [7] using the method chunks identified in Sect. 3.

Table 7. Defining OSS adoption business processes method chunk

Name	MC14: Defining OSS adoption business processes
Situation	Goal model representing organizational goals
Intention	Defining business processes for OSS adoption
Process part: 1. For each chunk with intention *Defining OSS adoption business processes* (MC4 – MC13), check if the organizational goals include a goal matching with the situation of the method chunk. 2. If there is such a goal apply the method chunk.	***Product part:*** BPMN diagrams imple-menting the OSS adoption strategic goals of the organization.

4.1 OSSAP Method Requirements Specification

The purpose of the OSSAP method is, first, to help organizations to refine their organizational goal models following an adequate OSS adoption strategy and, then, complement this with the OSS business processes describing the activities that the organization should undertake to implement the adoption strategy selected. In fact this is quite close to the intentions uncovered in Sect. 3.1 for the DKE-approach, therefore we decide to extend its process map (see Fig. 1) with the intention of obtaining OSS-aware business processes.

The final process map of the OSSAP method is illustrated in Fig. 2; plain lines indicate the intentions and strategies inherited from the DKE-approach, while dashed lines represent the new requirements. Only one new intention has been elicited: *Define OSS-aware business processes* (I4), and two new strategies: *Goal-elicitation techniques by reuse* (S1b), complementing the existing goal-oriented method to achieve I1; and *Process modelling (BPMN)* (S4) to achieve the new intention I4. The next subsection describes the chunk selection for these new strategies.

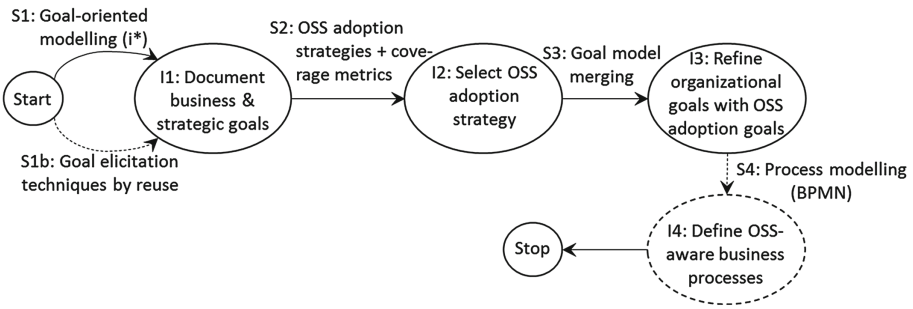


Fig. 2. Process map of the OSSAP method

4.2 OSSAP Method Chunks Selection

For the *Goal-elicitation techniques by reuse* strategy, we have selected to use as method chunk the set of *Business & OSS goals catalogues* (MC15) presented in [14]. OSSAP uses two of such catalogues: (1) the generic business goals catalogue, related to the external environment and the strategic organizational components (e.g., to consolidate market position); (2) the generic OSS goals catalogue, related to OSS adoption goals that any organization might want to achieve independently from the adoption strategy chosen (e.g., to avoid vendor/consultant lock-in).

For achieving the new intention I4 of defining OSS business processes, we use the new chunks created for this purpose described in Sect. 3.2 which already use BPMN as process modelling technique.

4.3 OSSAP Method Chunks Assembly

In the assembly-based SME approach, there are two assembly strategies: *association* and *integration* [7]. Association is used when the method chunks to assemble do not overlap in terms of intention to achieve and product to construct, for example when the results of one chunk are used as an input in the other. Integration is used when the chunks have similar engineering goals and their product models overlap.

The existing *MC1: Goal modelling with i** (see Sect. 3.1) and the new *MC15: Business & OSS goals catalogues* (see Sect. 4.2) are the method chunks selected for the strategies that reach I1. They share the same engineering goal, namely eliciting organizational goals; in addition, since the process of elicitation and documenting goals can be an iterative process, both can be combined and used in indistinct order. In this context, the integration strategy has to be used because both chunks contain the concept of goal (in the product part) and goals in the catalogues can be used in the *i** models as goals (or softgoals). It consists in simple merging of the common concepts; no naming problems have been identified.

On the other hand, the method chunks MC1 – MC3 selected for the strategies that attain the first three intentions (I1, I2 and I3) produce *i** goal models while the method chunks MC4 – MC14 selected to attain I4 produce BPMN models. Hence, we have two kinds of models: *i** and BPMN focusing on different, complementary aspects of an organization. These method chunks deal with complementary engineering goals and the simple association strategy is sufficient to assembly them, which consist in identifying links between concepts of different method chunks and ordering method chunks application. In OSSAP we consider that all the processes in business process modelling are defined to achieve a specific goal. Therefore, we need to create an association between process and goal concepts to establish the link between the selected OSS business process and the corresponding goal in the *i** model.

5 OSSAP Application: The TEI Case

We present the application of the OSSAP method to Ericsson Telecomunicazioni Italy (TEI). TEI is a division of Ericsson, one of the world's leading telecommunication corporations. One of TEI's roles within the Ericsson ecosystem is providing knowledge

and expertise on OSS alternative to support efficient third party product handling. All organizational processes in TEI are defined in a detailed way and thus the rigour of OSSAP is well-suited to the company. OSSAP can help TEI in being aware about which processes they need to embrace according to their strategic needs when using an OSS component instead of proprietary software.

According to the OSSAP process map (Fig. 2), the application to the TEI case has been divided on the achievement of the four intentions reported below. The first three are only briefly described since they have been presented in detail in [5]; still we include them to make the paper self-contained.

- **Intention 1. Document TEI business and strategic goals**. We apply the method chunk *MC1: Goal modelling with i** in order to obtain the TEI organizational model as starting point. A significant except of this model appears in [5].
- **Intention 2. Select the TEI OSS adoption strategy**. We apply the method chunk *MC2: OSS adoption strategy selection*. From its sub-chunks, TEI selects *MC2.2: OSS integration adoption strategy*. [4] presents the full implementation of this chunk, applying the coverage metrics defined therein.
- **Intention 3. Refine TEI organizational goals model with the selected strategy.** We apply the method chunk *MC3: i* model merging* in order to refine the documentation of the TEI organizational goal model. Figure 3 shows a significant excerpt of this model (different from the one presented in [5]).
- **Intention 4. Define the OSS-aware TEI business processes.** We apply the aggregate method chunk *MC14: Defining OSS Adoption Business Processes* to select the adequate sub-chunks. In Table 8 we list the goals in the refined TEI organizational model that have led to a selection together with the chunks selected.

For instance, one of the intentional elements of the TEI organizational model was *Integrate* as a means to use an OSS component integrating it in a software product (G3 in Table 8 and also one of the intentional elements appearing in Fig. 3). This goal matches the situation of two different chunks that provide business processes for two cases of implementing OSS integration: *Integrating and redistributing OSS* and *Integrating without redistributing it*. The business processes for these two cases are different because there are legal implications regarding OSS licenses that must be dealt differently when the adopter wants to redistribute the software. If the software is not redistributed, license compliances issues may not have to be checked. Actually, depending on the contextual information and business scenario, TEI applies any of the chunks related to using an OSS component (goal *OSS component used* in TEI goal model): sometimes they need to supply an OSS operating system (*Deploying OSS* chunk), or use OSS libraries to be included in their software systems (*Integrating and redistributing OSS* chunk), or use some OSS components to be integrated in the software they use internally (*Integrating OSS without redistributing it*). Another intentional element in TEI organizational model was *Develop patches* (G5 in Table 8 and also one of the intentional elements appearing in Fig. 3) because it is a means to contribute to the OSS community that helps the OSS component evolve towards the features desired by TEI. It matches the *Patching OSS* method chunk (described in detail in Sect. 3). The effect of this method chunk application will be that the Table 6 business process diagram will be incorporated to TEI business processes in order to

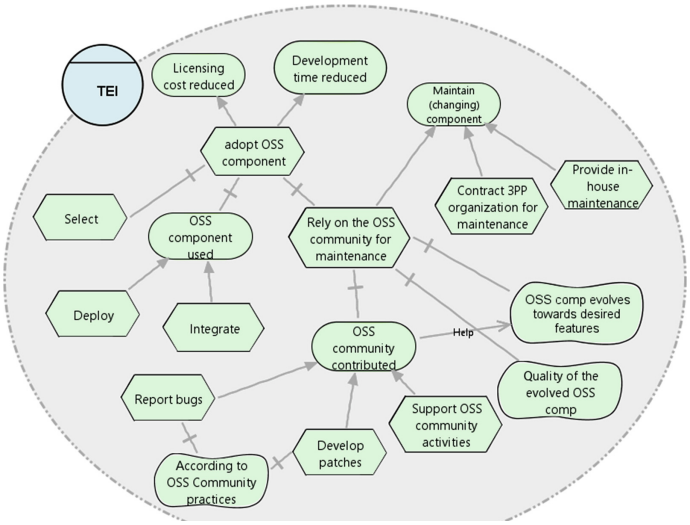


Fig. 3. Excerpt of TEI's organizational model adhering to the OSS integration adoption strategy

Table 8. Application of defining OSS adoption business processes (MC14) to the TEI case

Goal	*Method chunks selected*
G1: Select	MC5: Selecting OSS
G2: Deploy	MC6: Deploying OSS
G3: Integrate	MC7: Integrating and redistributing OSS
	MC8: Integrating OSS without redistributing it
G4: Report bugs	MC9: Reporting bugs about OSS
G5: Develop patches	MC10: Patching OSS
G6: Support OSS community activities	MC11: Supporting OSS community

implement the *Develop patches* intention. When TEI implements the *Patching OSS* process (see Table 6), since they are not interested on making public their code, they always go through the activity *Discuss solution* in the process defined by the chunk.

6 Discussion

In this section we analyze the relationships existing between the new method chunks for obtaining OSS business processes and the six OSS adoption strategies from the DKE-approach [5]. Since the method chunks have been identified from the goals of the OSS adoption strategies (see Sect. 3.2), clear relationships exist between them as can be seen in Table 9. Columns correspond to adoption strategies and rows to OSS business process method chunks. A dark cell means that the method chunk is mandatory to implement one of the goals of the adoption strategy (e.g. a fork strategy implies creating an OSS community in all cases). A grey cell means that the adoption

Table 9. Method chunks for OSS adoption strategies (black cell: mandatory, grey: optional).

Method chunks for defining OSS business processes	*Acquisition*	*Integration*	*Initiative*	*Takeover*	*Fork*	*Release*
MC4: Creating OSS			■			■
MC5: Selecting OSS	■	■		■	■	
MC6: Deploying OSS	▒	▒	▒	▒	▒	▒
MC7: Integrating and redistributing OSS	▒	▒	▒	▒	▒	▒
MC8: Integrating OSS without redistributing it	▒	▒	▒	▒	▒	▒
MC9: Reporting bugs about OSS		■	■	■	■	
MC10: Patching OSS		▒	■	■	■	
MC11: Supporting OSS Community		▒	■	▒	▒	
MC12: Leading OSS Community			■	■	▒	
MC13: Creating OSS Community			■		■	

strategy may require or not that method chunk (e.g. an integration strategy may require patching OSS or not). This optionality comes from the fact that, for some adoption goals, there are several business processes that can be used to achieve them.

Beyond pure engineering aspects, it is also worth mentioning the conceptual difference between the DKE-approach and the OSSAP method. Whilst the DKE-approach assumed that the OSS adoption strategies behaved as a kind of high level patterns to be applied in all organizational contexts, the situational nature of OSSAP recognizes the fundamental diversity that may exist in each and every OSS adopter organization. As Table 9 shows, too many aspects exist that are configurable in every strategy. This is why we consider OSSAP a step beyond the real context in OSS adoption. Still, the work done while designing the DKE-approach has been crucial to generate OSSAP. We may sense that the formulation of OSSAP starts a second cycle in a design science approach [15] after the validation done in practice of the former DKE-approach.

7 Conclusions and Future Work

In this paper we have proposed a method for defining OSS Adoption business Processes (OSSAP). It has been designed using the assembly-based situational method engineering (SME) approach. Applying SME allows us to reuse the existing method presented in [5] (DKE-approach) and complementing it with a set of new chunks defining business process in BPMN related to OSS adoption. The process model of OSSAP is formalised using the Map formalism. This map proposes four intentions and several strategies to achieve them. The first three intentions embody the selection of the OSS adoption strategy that best fits with the organization's goals, and the last one aims to identify business processes to fulfil them. The main contributions of this work are:

(1) The OSSAP method, which allows us to derive OSS-aware business process models from the combination of the starting organizational model and the OSS adoption strategy chosen and (2) A set of method chunks that can be reused in contexts other than OSSAP. They are general-purpose, e.g. the *i** framework method chunk, or domain-specific, as the set of method chunks for the adoption strategies.

Using SME for building OSSAP facilitates its extension. If new strategies for OSS adoption emerge, OSSAP could integrate them as new method chunks. In addition, OSSAP addresses the definition of business processes related to OSS adoption but the approach could be generalized to other kinds of processes, e.g., quality assurance.

To our knowledge, in spite of the huge OSS body of knowledge, this is the first attempt to systematically embody the consequences of OSS adoption into organizational business processes. Other approaches that analyse OSS adoption as for instance Chang et al.'s [16], Daffara's [17] and Dornan's [18] provide classification criteria for OSS business models that rely on the concrete way in which OSS components are adopted in the organization. However, they do not make any attempt to systematically describe the business processes implied by these adoption strategies (they are discursive papers) and do not link these processes to intentions or goals.

The TEI example of application has been used as a preliminary validation of the applicability of OSSAP. As it was mentioned in [5] related to the first part of the method (selecting the OSS adoption strategy), independently of the complexity of the organizational models, the portion of these models involved in the selection of the OSS adoption strategy are not expected to grow in a way that they will be unmanageable. On the other side, the number of identified business processes is quite small, allowing us to keep the level of complexity of their selection low. Of course, further validation or this statement is required.

Future work addresses the validation of OSSAP in other OSS adopter organizations in order to properly finalize this design cycle. Also we will analyse the possibility of making the process maps more abstract in order to explore other possible strategies for implementing their intentions. Therefore, we could substitute the selection of techniques in the strategies (*i**, BPMN and reuse-based elicitation) and leave room for other method chunks as KAOS [19], SPEM [20] or GRAM [21], respectively.

Acknowledgments. This work is a result of the RISCOSS project, funded by the EC 7th Framework Programme FP7/2007-2013, agreement number 318249. It was also supported by the Spanish project EOSSAC (TIN2013-44641-P).

References

1. Driver, M.: Hype cycle for open-source software. Technical report, Gartner (2013)
2. Henderson-Sellers, B., Ralyté, J., Ågerfalk, P., Rossi, M.: Situational Method Engineering. Springer, Heidelberg (2014)
3. Yu, E.: Modelling strategic relationships for process reengineering. Ph.D. thesis, University of Toronto, Toronto, Ontario, Canada (1995)
4. Object Management Group (OMG): Business process model and notation (BPMN), version 2.0. Technical report, January 2011

5. López, L., Costal, D., Ayala, C.P., Franch, X., Annosi, M.C., Glott, R., Haaland, K.: Adoption of OSS components: a goal-oriented approach. Data Knowl. Eng. **99**, 17–38 (2015)
6. Henderson-Sellers, B., Ralyté, J.: Situational method engineering: state-of-the-art review. J. Univers. Comput. Sci. **16**(3), 424–478 (2010)
7. Ralyté, J., Rolland, C.: An assembly process model for method engineering. In: Dittrich, K.R., Geppert, A., Norrie, M. (eds.) CAiSE 2001. LNCS, vol. 2068, pp. 267–283. Springer, Heidelberg (2001)
8. Ralyté, J., Rolland, C.: An approach for method reengineering. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) ER 2001. LNCS, vol. 2224, pp. 471–484. Springer, Heidelberg (2001)
9. Rolland, C., Prakash, N., Benjamen, A.: A multi-model view of process modelling. Requirements Eng. J. **4**(4), 169–187 (1999)
10. Ralyté, J.: Towards situational methods for information systems development: engineering reusable method chunks. In: Proceedings of the International Conference on Information Systems Development (ISD 2004), pp. 271–282 (2004)
11. Rosen, L.: Open Source Licensing. Prentice Hall, Upper Saddle River (2004)
12. Ayala, C., Costal, D., Franch, X, Franco, O.H., López, L., Morandini, M., Siena, A.: D1.3 Modelling support (Consolidated Version). Technical report, RISCOSS FP7 project (2014)
13. Mirbel, I.: OFLOSSC, an ontology for supporting open source development Communities. In: Proceedings of the 11th International Conference on Enterprise Information Systems (ICEIS 2009), SAIC, pp. 47–52 (2009)
14. Tapia, L.M., López, L., Ayala, C.P., Annosi, M.C.: Towards an OSS adoption business impact assessment. In: Ralyté, J., et al. (eds.) PoEM 2015. LNBIP, vol. 235, pp. 289–305. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25897-3_19
15. Wieringa, R.: Design Science Methodology for Information Systems and Software Engineering. Springer, Heidelberg (2014)
16. Chang, V., Mills, H., Newhouse, S.: From open source to long-term sustainability: review of business models and case studies. In: Proceedings of All Hands Meeting, OMII-UK Workshop (2007)
17. Daffara, C.: Business models in FLOSS-based companies. In: Proceedings of the Open-Source Software in Economic and Managerial Perspective Workshop (OSSEMP 2007) (2007)
18. Dornan, A.: The five open source business models. Information Week (2008)
19. Dardenne, A., van Lamsweerde, A., Fickas, S.: Goal-directed requirements acquisition. Sci. Comput. Program. **20**(1–2), 3–50 (1993)
20. Object Management Group (OMG): Software & systems process engineering meta-model specification (SPEM), version 2.0. Technical report, April 2008
21. Antón, A., Potts, C.: The use of goals to surface requirements for evolving systems. In: IEEE Proceedings of the 20th International Conference on Software Engineering (ICSE 1998), pp. 157–166 (1998)