Applying Business Strategy Models in Organizations[1]

Lidia López, Xavier Franch

GESSI Research Group, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
{llopez, franch}@essi.upc.edu

Abstract. Increasing adoption of Open Source Software (OSS) in information system engineering has led to the emergence of different OSS business strategies that affect and shape organizations' business models. In order to obtain the specific organizational model for a concrete organization that is adhering to a specific OSS business strategy, we need to instantiate the general knowledge included in this business strategy. This paper describe the process in which this general knowledge is instantiated and define a set of operations over *i** models to implement the instantiation concept. Although conceived in the field of OSS, the approach is generalizable to any kind of business strategy.

Keywords: Business strategy; *i** framework; i-star; OSS; Open Source Software

1 Introduction

The key purpose of any business is to create value and to achieve revenues. The business model of an organization captures the ways and means how these goals can be achieved. Therefore, there is no organization without a business model, regardless of whether or not a company can explicitly describe it [1][2].

A business strategy describes the approach of a business to successfully compete with other businesses in a given market. The key goal of a business strategy is to outperform competitors. As Porter points out, "a company can outperform rivals only if it can establish a difference that it can preserve. It must deliver greater value to customers or create comparable value at lower cost, or do both" [3]. In order to leverage business strategies with a business model, it is necessary to reconcile both worlds from different perspectives. If we think about organizational modeling as one of these perspectives, this means using the same modeling methods to produce unified models.

In the context of organizations adopting OSS components in their business strategies and business models, the FP7 European project RISCOSS (www.riscoss.eu) defines a portfolio of OSS business strategies that embrace well-established goals, activities and resources that characterize the main aim of each strategy. These OSS business strategies are modeled using the *i** framework [4], which allows the connection of low-level goals referred to the business strategies and the high-level business goals connected to the business model.

[1] This work is a result of the RISCOSS project, funded by the EC 7th Framework Programme FP7/2007-2013, agreement number 318249.

In this paper, we are interested in how these business strategy models have to be adapted in order to be included in the organizational model. Once the applicability of one strategy in a given context is decided, its model can be instantiated with organization-dependent information to obtain the actual business strategy as applied in the organization, and linked to the business model.

The remainder of the paper is organized as follows. Section 2 introduces the OSS business strategies used as a possible example of application in the context of OSS component adoption. Section 3 develops the main contributions of the paper including the formal definition of the instantiation operations used to instantiate the business models. Last, Section 4 provides conclusions and future work.

2 OSS Business Strategies

RISCOSS defines a portfolio of strategies depending on the degree of integration of the adopting organization in the community around the OSS component to be adopted. These strategies vary from organizations that only take the component without establishing any collaboration with the community (OSS Acquisition) to organizations that take the control of the community for their own business purposes (OSS takeover), including intermediate strategies where the company takes profit from the component supporting this development in some way (e.g. OSS Integration).

To illustrate the instantiation process that we present in this paper, we have chosen the simpler OSS business strategy that corresponds to the OSS Acquisition (Figure 1).

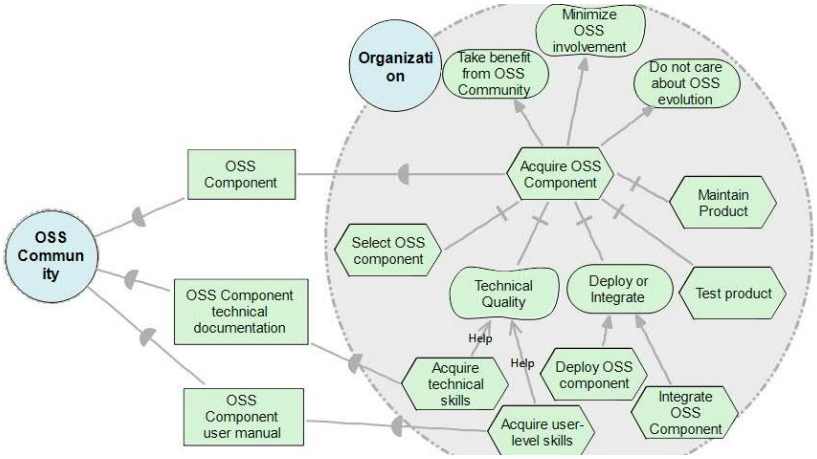

Figure 1: OSS Acquisition pattern.

3 Business Strategy Models in the Context of Organizations

We include some annotations to the *i** models to identify which model elements are candidates to be instantiated and which information should be used to this aim. Then, instantiation will be a process of assigning concrete values to these model elements.

3.1 Instantiating Business Strategy Models

Figure 2 shows what could be an excerpt of an organizational model, before applying an OSS business strategy model. The *i** model shows how the company ACME produces the product Road Runner Locator (RR Locator) for its customers. It consists of an *i** model with the high-level business goals and general activities of the adopter organization. The model reflects that the organization has made the decision of adopting an OSS component (a Geographical Information System) as a way to reduce costs following the OSS Acquisition strategy presented in the previous section[2].

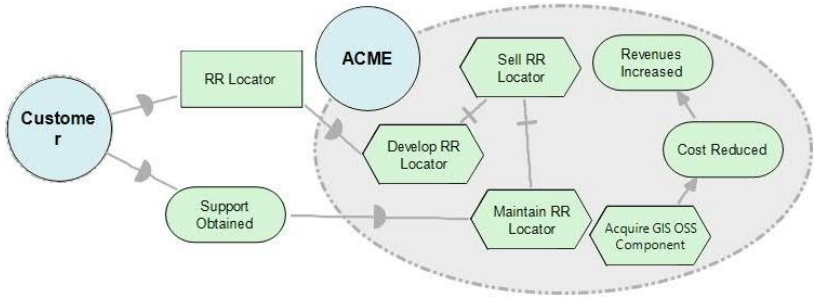

Figure 2. Initial organizational model

The business model is complemented including the elements of the OSS Acquisition business strategy model (Figure 1), as the decomposition to the task *Acquire GIS OSS Component*, and the missing actor *OSS Community* and dependencies *OSS Component*, *OSS Component technical documentation* and *OSS Component user manual* as shown in Figure 3.

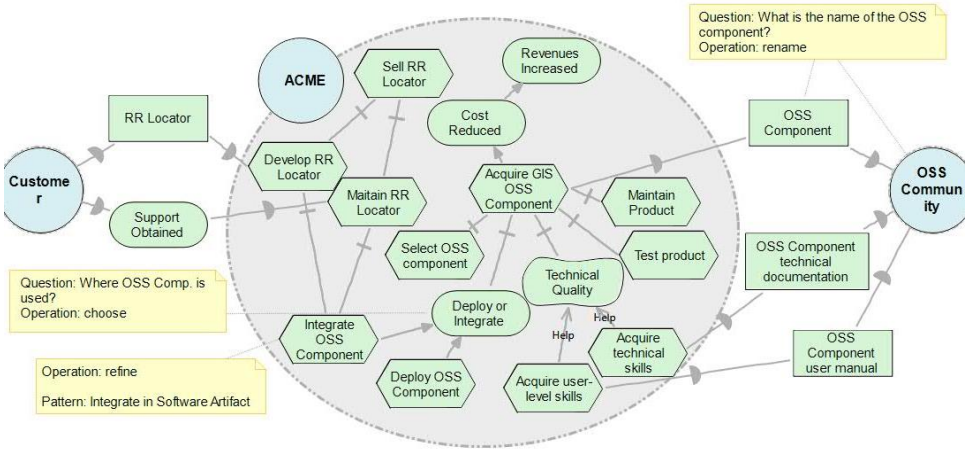

Figure 3. Organizational model with OSS Acquisition business model included

[2] The way of the OSS business strategy is chosen is out of scope of this paper.

The OSS business strategy models elements are annotated to indicate that they need some instantiations in order to consider the resulting model adapted to a certain organization. There are 3 kinds of annotations: 1) *rename*, used to give a concrete name to a generic name (e.g. what is the name of the OSS component to be adopted?); 2) *choose*, used to select among different options (e.g. will the OSS component be integrated or deployed); and 3) *refine*, used to mark an element as pending of further refinement (e.g. how will the OSS component be integrated into the product architecture?).

It is also worth to mention that the elements from the strategy business model may require tight integration with other parts of the model. Figure 3 shows how the task *Integrate OSS Component* is considered also a subtask of both *Develop RR Locator* and *Maintain RR Locator*.

Next, the part of the model coming the OSS Acquisition business model needs to be instantiated following the annotations. Figure 4 shows the result of instantiating the generic OSS Component by the *OSS GIS* (in the resource that represents the component and the community that produces it). Also the decision among integrate or deploy is made. Since integration (*Integrate OSS Component*) is chosen, the refinement of this task using further knowledge (e.g., we could have lower-level architectural integration models) should be undertaken next. This process of instantiation will be repeated until all the annotations are solved.

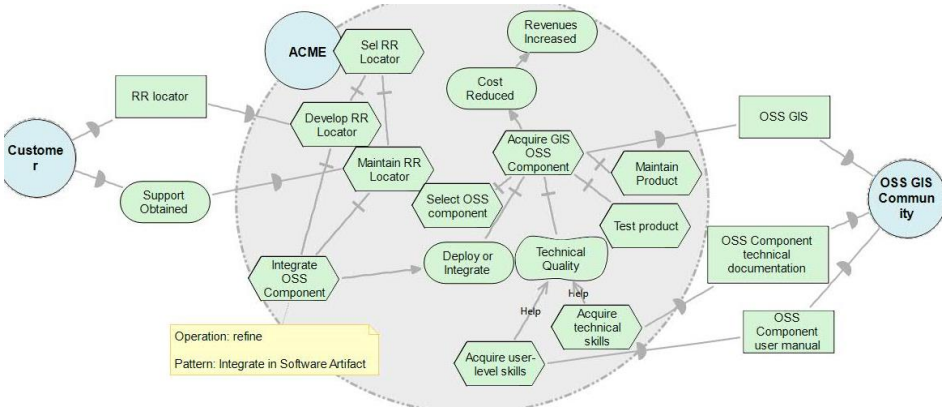

Figure 4. Instantiated model choosing the OSS Acquisition business strategy

3.2 Instantiation operations

This section outlines the formal definition of the instantiation operations. We have used the formalization of *i** presented in [5] as a basis of this definition. The general layout consists on defining elements as tuples of sub-elements and then functions with a meaningful name to obtain these sub-elements. In a nutshell, an *i** model (*M*) contains actors (*A*), dependencies (*DL*) and dependums (*DP*). Actors contain intentional elements (*IE*s, of type: goals, softgoals, tasks and resources) connected by IE links

(*IEL*) of different types. Dependencies connect actors and/or IEs inside them and have a dependum (that is also an IE). The elements are identified by their name (n_x). An example of function to obtain sub-elements is actors(*M*), defined as actors(*M*) = *A*.

Considering: an *i** model $M = (A, DL, DP)$; an actor $a = (n_a, IE_a, IEL_a)$, $a \in A$; an intentional element $ie = (n_{ie}, t_{ie})$, t_{ie} being the type of intentional element, $ie \in IE_a$; a dependency link $dl = (dr, de, dm)$ with dependum $dm = (n_{dm}, t_{dm})$, $dl \in DL$, Table 4 summarizes all the instantiation operations, whose complete definition cannot be included for space reasons. For renaming operations, *n* stands for the (non-empty) new name to be given to the renamed element. The new name cannot be already used by any existing element (see preconditions), e.g. *actor*(*A*, *n*) = Ø means that there is no actor named as *n* in the actor's set (*A*). *IE* stands for intentional element. *sources* represents the means to achieve an IE (means in a means-end or subtasks in a task-decomposition), *roots* return the roots of an actors' SR.

Table 1. Instantiation operations

Preconditions	**Resulting *i** model *M'* defined as**
***renameActor*(*M, a, n*)**	
$actor(A, n) = \emptyset$	$M' = substituteActor(M, a, (n, IE_a, IEL_a))$
***renameIE*(*M, a, ie, n*)**	
$intentionalElements(a, n) = \emptyset$	$M' = substituteIE(M, a, ie, (n, t_{ie}))$
***renameDependum*(*M, dl, n*)**	
	$M' = (A, DL \setminus \{dl\} \cup \{dl_{new}\}, DP \cup \{(n, t_{dm})\})$ where $dl_{new} = (dr, de, (n, t_{dm}))$
***chooseIELinks*(*M, a, ie, IES*)**	
$IES \subset sources(ie, IEL_a)$ $IES \neq \emptyset$, at least one kept	$M' = substituteActor(Mdep\text{-}, a, b)$ where $b = deleteIEDecomposition(a, sources(ie, IEL_a) \setminus IES)$ $Mdep\text{-} = deleteDependencies(M, sources(ie, IEL_a) \setminus IES)$
***refineIE* (*M, a, ie, ies, t, v*)**	
$l = (ie_s, ie, t, v) \notin IEL$ $ies \notin roots(b)_a$	$M' = substituteActor(M, a, b)$ where $b = (n_a, IE_a \cup \{ie_s\}, IEL_a \cup \{(ie_s, ie, t, v)\}, t_a)$
Additional functions	
substituteActor(M, a, b)	Returns *M'* substituting actor *a* in *M* by one *b*
substituteIE(M, a, ie, (n, t))	Returns *M'* substituting the IE *ie* in *a* at *M* by (*n*, *t*)
deleteIEDecomposition(a, IES)	Returns an actor removing the intentional elements *IES* from *a*, and additionally removing dangling links
deleteDependencies(M, IES)	Returns *M'* deleting the dependencies connected to the set of intentional elements *IES* from *M*

3.3 Tool support

This model instantiation is partially implemented by the GoalModels tool, included in the platform developed by RISCOSS, developed in Java, that uses the *i** mark-up iStarML [6] to store the models. This formatting language allows saving models textually, using XML specific tags and attributes to describe the different elements in the model. Although the primary focus of iStarML is model interchange, i.e. the provi-

sion of an interoperability standard, the tool uses it for implementing model persistence and uses XPath for querying the model.

The tool provides only 2 public functions: `execute(…)` and `questionnaireAnswered(…)` in order to start the process obtaining the first set of pending questions (`execute`) and process the answers for a set of questions and obtain the next ones (`questionnaireAnswered`). This last function is called iteratively until the list of pending questions becomes empty.

An aspect that needs to be mentioned is the need to extend the *i** language with the information required by the instantiation operations. This part was really easy since an iStarML characteristic is the simplicity to extend the format without affecting the rest of the model: new tags and/or attributes can be added and other iStarML users are not affected by this new information. The new information has been added as new attributes for some specific defined tags.

4 Conclusions and future work

In this paper we described part of the process to model organizations within the European project RISCOSS. We use *i** models to capture the intentional and business aspects that drive an organization to adopt an OSS component. In particular, we include the iterative process for constructing the organizational model for a particular organization from some pieces of generic models. The *i** models are annotated pointing the candidate parts of the model to be shaped depending on the organization. These annotations consist on what kind of shaping is applied (operation) to a specific model element and the information needed from the organization in order to perform the shaping (instantiation).

As part of the future work, we need further work in these operations including their validation executing the use cases defined in RISCOSS. At this point, we already identified a new operation needed to merge models, for the case of more than one strategy business applies at the same time. We also need working on a separate operation for matching the organizational model with the business strategy models in order to identify the potential strategies that would fit to the organization.

References

[1] Chesbrough, H.: Open Business Models: How to thrive in the new Innovation Landscape. Harvard Business School Press, 2006.

[2] Teece, D. J.: Business Models, Business Strategy and Innovation. In Long Range Planning Journal, Vol. 43(2-3), 2010.

[3] Porter, M. E. What is Strategy? Harvard Business Review, Nov.-Dec. 1996.

[4] Yu, E.: Modelling Strategic Relationships for Process Reengineering. PhD. thesis, Toronto, 1995.

[5] Lopez, L., Franch, X., Marco, J.: Specialization in *i** Strategic Rationale Diagrams. ER 2012. LNCS Vol. 7532, 2012.

[6] Cares, C., Franch, X., Perini, A., Susi, A.: Towards Interoperability of *i** Models Using iStarML. In Computer Standards & Interfaces Journal. Vol. 33 (1), 2011.