Using Roles for OSS Adoption Strategy Models

Dolors Costal, Lidia López, Xavier Franch

GESSI Research Group, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
{dolors, 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 adoption strategies that affect and shape organizations' business models. OSS adoption strategies can be operationalized by *i** models describing the consequences of choosing each strategy. When an organization decides to adopt an OSS component, it becomes a part of the OSS ecosystem around this component. Therefore, OSS adoption strategy models need to be structured in the way to explicitly describe the role of the adopter organization within the OSS ecosystem, which may be quite different depending on the level of compromise that the organization prefers. Making visible the roles played by the different agents involved in the OSS ecosystem, the involvement of the organization in the OSS community arises naturally. This paper includes a set of roles that emerge in an OSS ecosystem and their responsibilities, and describes the issues behind the fact of using the *i* role* and *plays* constructs.

Keywords: Open-Source Software; OSS; OSS adoption; OSS ecosystem; i-star; organizational role.

1 Introduction

In the context of organizations adopting OSS components in their business strategies and business models, [1] defines a portfolio of OSS adoption strategies that embrace well-established goals, activities and resources that characterize the main aim of each strategy, as part of the FP7 European project RISCOSS (www.riscoss.eu) [2]. These OSS strategies are modeled using the *i** framework [3], which allows the connection of low-level goals referred to the adoption strategies and the high-level business goals connected to the business model.

The OSS adoption strategies models contain two actors: (1) the organization that adopts the OSS component (OSS Adopter) and (2) the OSS community that produces it. The adoption strategies are defined in order to describe the role of the OSS adopter in the OSS Community, i.e. in the OSS ecosystem around the OSS component. The rationale behind the elements inside the organization actor is identifying the goals and activities that the company needs to achieve and perform connected to the role of the organization in the OSS ecosystem. Most of the goals and tasks inside the organization actor are included because the organization becomes part of the community. This fact indicates that they should be in the OSS community actor in some way instead of in the adopter organization. The OSS Community actor should be refined making the different roles explicit.

Making these roles explicit, we would take advantage of the *i** modelling language highlighting the motivation for having some elements in the model. If an organization (*OSS Adopter*) wants to contribute to the community (goal *OSS community contributed*), it can do it for example producing code (*Developer*). Therefore, the task *maintain code* must be included in the OSS adopter rationale (Fig. 1, left side). Using roles would explicitly add the rationale for the presence of some activities, for example, the OSS Adopter needs to *maintain code* only because the organization has decided to contribute to the OSS community playing the role of *Developer*.
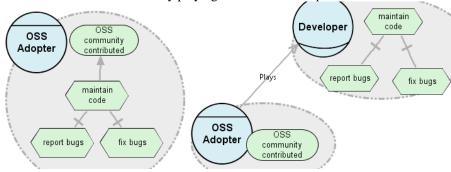

Fig. 1 OSS adoption strategy models using (right) and not using (left) *i** roles[1]

The OSS adoption strategies models catalogue presented in [1] contain all the activities modelled inside the rationale of the *OSS Adopter* actor according to its adoption strategy. That paper also uses the notion of coverage to assess which is the OSS strategy that better fits the organizational goals. The use of roles has been discussed regarding understandability versus modularity; the proposal presented in [1] prioritizes understandability (model on the left). In the work presented in this paper, we explore the use of roles in the definition of OSS adoption models, prioritizing modularity. Using roles also helps to understand the responsibilities assigned to each role, but it increases the complexity and makes the model less legible. Fig. 1 shows both alternatives. The use of roles for representing the responsibilities associated to a user group (OSS Adopters in our case) is also present in previous works like [4], where roles and the *plays* link are used to encapsulate and analyse the role's (e.g., Developer) responsibilities that are present in the agent playing the role (e.g., OSS Adopter).

The remainder of the paper is organized as follows. Section 2 introduces the OSS roles that characterize an OSS Community. Section 3 develops the main contributions of the paper including the proposal of using *i** roles in order to structure the OSS adoption strategy models. Last, Section 4 provides the open issues to be considered during the research in order to complete our proposal.

---

[1] The names of the intentional elements inside the actors have been modified respect to the names in the OSS adoption strategy models from [1] to improve the legibility of the models.

2   OSS Roles and Activities Ontology

In the context of the European Project RISCOSS, we have developed an ontology for supporting risk analysis in OSS ecosystems [5]. Part of this ontology contains terms for describing OSS projects. The activities and resources relevant for the interaction between an OSS adopter and an OSS community has been used for creating a set of models describing different strategies that the organizations can follow when they need to adopt an OSS component [1]. In [1] we have presented in detail the terms of the RISCOSS ontology related to the activities and resources used for developing the OSS adoption strategy models. The work presented in this paper is related to the terms defining the different roles in the OSS projects.

Table 1 contains an excerpt of the OSS roles defined in the RISCOSS ontology, for each role there is the agent than can play it (property *played by*), the decision that it can take (property *take*) and the activities that it can perform (property *perform*). The resources involved in the activities are included into brackets. Indentation in the OSS Roles column indicates the existence of a category/subcategory relationship between roles.

Table 1. OSS Roles

OSS Roles	Properties
Governance Role	
CommunityManager	played by Visionary
Communicator	played by Communicator
Contributor	
Administrator	take MediumTermDecision, perform MaintainSite
Developer	perform MaintainCode, FixBug, DiscussBug (BugMessage), DiscussChange (ChangeMessage)
Committer	take ShortTermDecision, perform Supportnewbies, DiscussSolution (SolutionMessage)
Disseminator	played by Distributor, perform PackageFlossForNewUsers
Documenter	perform MaintainDocumentation
Project Manager	take LongTermDecision, played by Visionary, perform Nominate
Tester	perform DiscussBug(BugMessage)
User	perform learn using OSS, ask questions

3   Using Roles to Model OSS Adoption Strategies

When an organization decides to adopt an OSS component, it may have different levels of involvement in the OSS community depending on its adoption strategy [1]. It may have a minimum involvement of only using the OSS component (OSS Acquisition and OSS Release); it may be actively involved in the OSS community by performing some activities such as reporting bugs, developing patches, etc. (OSS Integration and OSS Fork), with the purpose of gaining influence on the OSS component

evolution; or, even, it may be deeply involved in the OSS community, trying to lead it (OSS Initiative and OSS Takeover). These different levels of involvement can be modelled in a natural way using the *i* role* and *plays* constructs. More concretely:

1) As a first step, we use *i** roles to model the set of OSS roles described in Section 2 (see Table 1).

2) Then, as a second step, we use the *i** plays construct to relate an *i** agent representing an OSS adopter organization with the adequate *i** roles obtained from the first step. To choose the adequate roles for an OSS adopter organization, we take into account the activities that it performs in the OSS community according to its adoption strategy. The result of this second step is the OSS adoption strategy model of the organization.

Next two subsections illustrate these two steps.

3.1   OSS Roles as *i** Roles

i* roles can be used to model the set of OSS roles described in Section 2 (see Table 1) and the relationships that exist between them. Fig. 2 represents these roles and their specialization (is-a) and aggregation relationships (is-part-of). To complete the roles definition, we add the activities that each OSS role performs represented as *i** tasks in the corresponding *i** role SR diagram (not shown in Fig. 2 for space reasons). For example, the role *User* has as tasks: *learn using OSS, ask questions*.
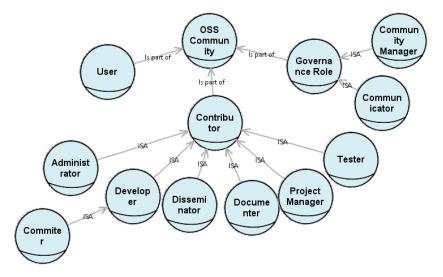

Fig. 2 OSS roles as *i** roles

The complete OSS ecosystem model, including dependencies between roles, has been obtained by using an adaptation of the RiSD methodology presented in [6] for systematic construction of i* SD models for ecosystems.

3.2   Use of *i* Roles* and *i* Plays* for OSS Adoption Strategy Modelling

Once we have the *i** roles and their SR diagrams, the *i** plays construct can be used to relate an agent representing the OSS adopter organization to them. For instance, if we have an OSS adopter whose adoption strategy is OSS Acquisition, which consists on having a minimum involvement in the OSS community meaning that it basically wants to use the component, this adoption strategy can be simply modelled by means of defining a plays relationship between the agent that represents the OSS adopter and the *User i** role (no more role are played by the adopter in this strategy). Fig. 3 illustrates this case.
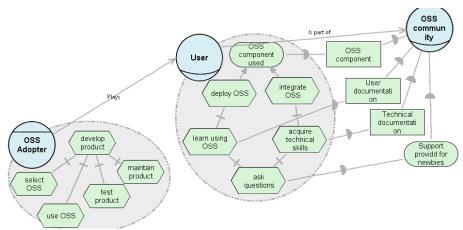

Fig. 3 Using *i* plays* construct to relate the OSS adopter and User

4   Open Issues

There are some open issues related to the presented proposal of using roles to model OSS adoption strategies.

First, the implications of using the *i** plays construct arise some questions over its related agents and roles that need clarification. For example: Does the agent that plays a role inherit all the dependencies defined on the role? Are the agent's goals exactly the same as those of the role? May the agent have additional goals? May the agent get rid of some role's goals?

Second, we have used the is-part-of relationship to relate, for example, the *User* role and the *OSS community* role (see Fig. 2 and Fig. 3). Therefore, when applying the adoption strategy model to a specific OSS community, e.g. the *Eclipse community*, we would need to replace *OSS community* by *Eclipse community* in the model and, thus, represent that the *User* role is part of the *Eclipse community*. However, this is not straightforward, since the is-part-of relationship is required to relate only actors belonging to the same type [7], and we have that *User* is a role while the *Eclipse community* is an actor that seems more realistic to model as an agent than as a role.

Finally, not all the goals of an OSS adopter, according to the [1] catalogue of OSS adoption strategy models, can be located as goals of the OSS roles (roles depicted in Fig. 2) because not all the adopters that will apply a specific role will share them. For instance, considering the OSS Acquisition adoption strategy defined in [1]:

1) The acquisition adoption strategy includes the following goals for the OSS adopter organization: *Take benefit from OSS Community*, *OSS involvement minimised* and *Do not care about OSS evolution for maintenance*.

2) The role played by an adopter applying the acquisition strategy would be that of *User* (it will apply only this role).

3) The *User* role is played also by adopters applying other adoption strategies such as, for example, the OSS Integration adoption strategy [1].

4) The goals listed in (1) cannot be placed in the *User* boundary since not all the adopters playing the *User* role will share those goals. In particular, adopters applying the integration strategy do not share the goals *OSS involvement minimised* and *Do not care about OSS evolution for maintenance.*

Acknowledgements

References

[1] López, L.; Costal, D.; Ayala, C.; Franch, X.; Annosi, M.C.; Glott, R.; Haaland, K.: Adoption of OSS components: A goal-oriented approach. In Data & Knowlegde Enginering. 2015. http://dx.doi.org/10.1016/j.datak.2015.06

[2] Franch, X., et al.: Managing Risk in Open Source Software Adoption. In International Conference on Software Engineering and Applications (ICSOFT 2013), pp. 258-264.

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

[4] Sun, J.; Liu, F.; Zhang, H.; Liu, L.; Yu, E.: Understanding the Diversity of Services Based on Users' Identities. In International Conference on Advanced Information Systems Engineering (CAiSE 2011), pp. 612-626.

[5] C. Ayala, L. López, D1.3 Modeling Support (consolidated version), Technical Report, RISCOSS FP7 project, 2014.

[6] Franch, X., Grau, G., Mayol, E., Quer, C., Ayala, C., Cares, C., Navarrete, F., Haya, M., Botella, P.: Systematic Construction of i* Strategic Dependency Models for Sociotechnical Systems. In International Journal of Software Engineering and Knowledge Engineering (IJSEKE), 2007, Vol. 17(1), pp.79-106

[7] *i* Guide*, ISA Association, istarwiki.org