Towards Extension Mechanisms in iStar 2.0

Enyo Gonçalves[1,3], João Araújo[2] and Jaelson Castro[3]

[1] Universidade Federal do Ceará, Brazil
[2] Universidade Nova de Lisboa, Portugal
[3] Universidade Federal de Pernambuco, Brazil
enyo@ufc.br, joao.araujo@fct.unl.pt, jbc@cin.ufpe.br

Abstract. iStar has been extended since its initial proposal in the 90's. In a previous work we have identified that 96 extensions had been proposed until 2016. It is worth noting that since 2016 the language notation is under standardisation. However, new extensions continue to be proposed. So, it is essential to discuss extension mechanisms to be included in the new standard. Hence, in this paper, we present some preliminary results related to this topic. We performed an exploratory study to analyse previous extensions and identified patterns of light-weight representations. The results of this study point out to 8 objectives and their representations. Furthermore, we used a survey to identify the opinion of 12 experts about this theme. Most of the participants considered important to include extension mechanisms in iStar, and indicated the light-weight representations of extensions that should be considered in this proposal. Finally, we discuss possible ways to propose the extension mechanisms in iStar 2.0 and present a preliminary proposal of these mechanisms.

Keywords: Requirements, iStar 2.0, Extension Mechanisms.

1 Introduction

Several iStar extensions have been proposed since the proposal of Yu [11] in 1995. In a previous work, we performed a Systematic Literature Review (SLR) [6] which identified 96 extensions proposed until 2016. The extensions proposed were related to several domain/application areas such as Social-Technical Systems (STS), Intelligent Agents, Software Product Lines, Legal Systems and others. Part of the extensions is widely used by the iStar community to model systems or are the basis for other extensions. We can cite TROPOS [3], GRL [1] and Secure Tropos [5] as examples of extensions widely used as the basis for the proposal of other extensions. Extension mechanisms are important for modelling languages as they allow their extensions without changing the respective metamodels. iStar is currently under standardisation [4]. Hence this is a suitable moment to discuss how extension mechanisms can be included in iStar 2.0. This paper presents an analysis of the light-weight representations in existing iStar extensions, the results of a survey with experts in iStar extensions and a preliminary proposal of extension mechanisms to be considered for inclusion in the new version of iStar 2.0.

2 Methodology

This paper presents results of three studies, two of them analyses the existing iStar extensions and the last one proposes a preliminary version of extension mechanisms. The first study (Section 4.1) identifies the light-weight representations used in existing iStar extensions and we counted the occurrence of each one to show a ranking of these representations.

The second study (Section 4.2) is a survey which the main goal was to evaluate which light-weight representations of the first study (presented in Table 1) should be included as standard extension mechanism in iStar 2.0 and if the language should have extension mechanisms. We followed the principles of Survey Research proposed in [9]. This survey is cross-sectional. The application takes place through a self-administered questionnaire via the internet, since the participants were in different countries. We invited to participate in the survey the experts who participated in a previous qualitative study [7] and with more iStar extensions according to the results of top authors in our SLR [6]. Clarification and consent terms were sent to participants with the invitation to participate in this research. So, the survey was composed of evaluation of 10 objective questions. 9 of them were concerned with analysing if each light-weight representation presented in Table 1 should be included as part of the standard extension mechanism, while the last question asked if the iStar language should have extension mechanisms. At the beginning of the survey, we highlighted that we intended to understand which light-weight representations should be selected and the figures presented in the questionnaire are only illustrations to understand each light-weight representation of Table 1. The options of the questions were defined in Likert scale with the values: Strongly disagree (1), Weakly disagree (2), Neutral (3), Weakly agree (4) and Strongly disagree (5). We validated the survey testing it with eight PhD students in Computer Science. So, they answered the survey and send us by email comments to improve the survey. We made the changes suggested during the test, and the responses were not used in the survey results. The survey can be accessed at https://goo.gl/forms/OgNPhOh7nQZFF1fK2.

The third study (Section 4.3) is a preliminary proposal of extension mechanism based on a benchmark of other modelling languages and the results of studies 1 and 2 of this paper. The selected light-weight representations were considered to define the default properties in our proposal.

3 Related Work

Extension mechanisms have been proposed similarly to the User Requirements Notation (URN) [2 and 8] and Unified Modelling Language (UML) [10]. Some ideas of these works can be useful in our definition of iStar extension mechanisms.

URN is a modelling language that joined Use Case Maps (UCM) and GRL. URN is specified as an international standard [8] which defines all characteristics of the language. According to [2], URN allows profiling by metadata, URN Links, URN concerns and OCL constraints. So, the extensibility of URN is represented regarding

metadata and URN links, which enables the support of most of the extensions proposed in the language, except for new graphical representation. Metadata are represented by name/value pairs to represent additional information in URN models. URN Links are used to specifying relationships between any pair of URN elements. Additionally, URN concerns are used to group any set of URN concerns, and OCL constraints can be used to define rules applied to the extension.

For a long time, UML had three extension mechanisms: stereotypes, tagged values and constraints. These mechanisms are textually represented, as stereotypes are represented between << >> and tagged values and constraints are represented between {}. The profile diagram was included recently in UML, and it is used to define profiles based on the extension mechanisms which are applied to the models during modelling time. The UML specification cites that in theory the profiles capability can be used to define extensions for metamodels other than UML.

4 Results

4.1 Identified Patterns of Light-weight Representations

We analysed the existing iStar extensions in [6]. In this previous work, we identified individually each representation for each light-weight representation and the number of the extensions which use each one. In a complementary way, our objective in this section is to identify the number of constructs and extensions for each light-weight representation resulting and doing so to generate a ranking for them (Table 1). This information is used in sections 4.2 and 4.3. So, we can identify patterns in the light-weight representations of additional information in the models. Note that new nodes with new graphical representations do not represent any pattern given that each new concept requires a new representation. Table 1 shows the number of constructs and extensions of each Light-weight representation. Specialised Node is used to specialising an existing entity to represent a new concept. It can be represented between <<>> or as a textual label added to an existing node. Thus, Specialised Link is used to specialise an existing relationship, to represent a new relationship.

The purpose of Node Identifier is to identify a goal/quality/task/resource with a short label that is used in some situations: i) it is used together with a reasoning technique; ii) also to make it easier to establish a reference to a goal/quality/task/ resource in a diagram that is part of a requirements document; or iii) to help to define priority of entities.

Node Status adds the status of a goal/quality such as denied, achieved and maintained. Label with syntax is used to add an additional representation using a kind of logic syntax in a textual label in diagrams. Cardinality is used to represent the number of elements in an iStar model. E.g., a cardinality <1..1> is used in means-end links to represent or exclusive or the cardinality [0..1] is used in Resource to represent an optional resource and [1..1] to represent a mandatory task. Reference to external representation is used to link an iStar diagram to external information. Reference to another part of the diagram is used to improve the modularity and scalability of iStar

models. It links parts of iStar diagrams. Link Qualifier is used to qualify a relationship; the relationships help, break and hurt specialise it.

In general, there is no unique kind of representation for each light-weight representation informed. So, Node identifier, e.g., is represented by five different ways: *Textual label without a special marker*, *Textual label between "[]"*, *Textual label before ":"*, *Textual label before "-"* and *Using a coupled graphical representation*. Further details about the representations used can be found at [6].

Table 1. Light-weight representations of iStar extensions.

Light-weight representations	Number of Constructs	Number of Extensions
Specialised Node	76	44
Specialised Link	63	43
Node Identifier	19	15
Node Status	14	9
Label with Syntax	13	9
Reference to External Representation	8	8
Cardinality	4	4
Reference to another Part of the Diagram	1	1
Link Qualifier	1	1

4.2 Survey with Experts in iStar Extensions to Select a Subset of Light-weight Representations

This section presents the results of the survey with experts to evaluate which light-weight representations of the first study (presented in Table 1) should be included as standard extension mechanism in iStar 2.0 and if the language should have extension mechanisms. The means of the responses are presented in the Fig. 1.

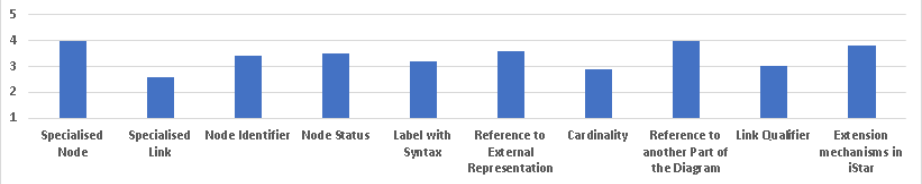

Fig. 1. Results of the mean of a survey about extension mechanisms in iStar.

Note that Specialised Link, Cardinality and Link Qualifier means are below or equal to 3. However, we believe that some mistakes may have influenced their opinions about *Specialised Link.* Four participants who evaluated *Specialised Link* as neutral, weak or strong reject commented that it is interesting to consider *Specialised Link* in the extension mechanism, but they did not agree with the representation used by us to illustrate this concept in the survey - for example we illustrated the specialised link by the usage of a label, we did not use "<<>>" such as stereotypes, and according to these participants, this representation is better. So, if we consider that this is the reason for the low evaluation, perhaps we can accept the *Specialised Link* as a possible part of the extension mechanisms definition.

Specialised Node, Node Identifier, Node Status, Label with Syntax, Reference to External Representation and *Reference to another Part of the Diagram* had the mean above 3. Remember that the value 3 represents neutral in the Likert scale used in this survey, so mean above 3 means that the related representation is important in the extension mechanism proposal.

Finally, the last question analyses if the iStar extensions should have extension mechanisms had value above 3. In another study [7] we analysed a set of statements using a survey, one of the statements was related to defining extension mechanisms. This statement received mean four according to the responses of other 30 experts in iStar extensions. The results of both studies reiterate the need for define extension mechanism.

4.3 Preliminary Proposal of the iStar Extensions Mechanisms

Node status is the fourth light-weight representation more used in extensions with fourteen constructs in nine previous extensions. So, it should be considered as a kind of representation frequently used in iStar extensions. Cardinality and link qualifier will not be considered here due to the value of its mean in the results of the survey and its low use in previous extensions (4 and 1 extensions, respectively).

We identified two ways to represent the extension mechanisms in iStar. The first one is to select one of the existing representations for each light-weight representation identified. So, we can have 8 different representations, one for each light-weight representation. For example, we can define that *Node Identifier* is represented by a ":" in the label of entities, Label with syntax represented by "{{}}" and so on. Another way is to consider the abstractions of UML and URN extension mechanisms as the base and adapt these ideas in the definition of iStar extension mechanisms. So, we think this second manner is better than the first one, since it is more generic and allows customisations.

So, the idea of UML stereotypes can be used to represent the specialisation of nodes and links in iStar. The UML tagged values have no similar representation in URN models, but we believe it is suitable in iStar context to represent the light-weight representations which we presented in Section 4.1 of this paper.

Additionally, default properties associated with tagged values could be defined. In UML there is a set of 33 default stereotypes such as *<<Utility>>*, applied to a Class to inform that it has no instances, but rather denotes a named collection of attributes and operations. So, we can define some default properties to tagged values to represent part of the light-weight representations selected in Section 4.2. Hence, *Node identifier* could be referred as *Id*, reference to external representation and reference to another part of the diagram can be adopted. They can be joined in a representation labelled as *Reference to*. *Node status* can be labelled as *Status*. Finally, Label with syntax can be labelled as *Logic*. Thus, a goal can be labelled, as for example as *<<Business>> {Id = G1} Accommodation booked*. The additional information can be used in a reasoning approach which can consider only business goals and take the value of *Id* (i.e., G1) of all business goals. The user could use the predefined iStar tagged-values and define their own. Fig. 2 presents an illustration of part of the exist-

ing representations and the representation with our proposal to *Node Identifier* and *Specialised Node*.

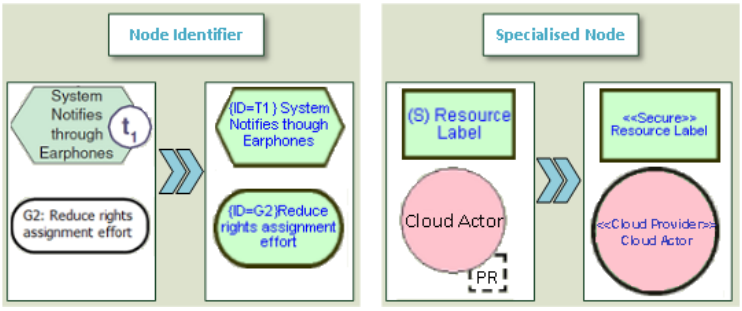

Fig. 2. Current Representations and Representation with the Extension Mechanisms.

The profiling is a characteristic present in UML and URN. The UML uses a specific model to allow the definition of UML profiles. It uses the class representations as basis. In URN the profiling is defined with the usage of metadata, URN links, URN concerns and OCL constraints (see Section 3). These elements are defined as properties of the elements of the models in the tool jUCMNav Tool. We believe URN concerns and OCL constraints could be adapted to become part of iStar extension mechanism regarding group concepts and define constraints between them.

In summary, we conclude that iStar could have the iStar stereotypes and iStar tagged-values visual light-weight mechanisms as a specialisation of iStar extension mechanisms. Additionally, it would have two new elements named iStar groupers and iStar OCL constraints, which could be hidden as properties in a modelling tool. The iStar groupers are useful to group metaclasses and make easy define constraints for a set of metaclasses in group.

These concepts should be part of the iStar syntax, so we represented them in the iStar metamodel. The metaclasses *Stereotype* and *TaggedValue* were associated with Element propagating these properties to all iStar nodes and links. We created an enumeration related to default tagged values (Id, Reference to, Status and Logic). The iStar groupers and iStar OCL constraints are represented as properties. The iStar metamodel with the representation of extension mechanism is available at http://www.cin.ufpe.br/~ejtg/istar_metamodel_extension_mechanisms/.

The gain with the iStar extension mechanisms is not only about simplification of the scenario presented in Fig. 2. Tools are relevant for the usage of an extension, but 51 extensions (53.6%) do not have any modelling tool [6]. Once these mechanisms become part of the iStar metamodel and the iStar modelling tools, a great part of future extensions could be proposed without coding tools, using only the extension mechanisms. So, they could contribute to providing tool support for iStar extensions.

5 Conclusions and Future Work

Many extensions have been proposed since the initial iStar version, in the 90's. At this moment iStar is under standardisation. Consequently, it is relevant to discuss iStar extensibility mechanisms. In this paper, we presented some preliminary considerations for extension mechanisms. It was based on analysis of existing extensions and considered the opinion of the experts about the selection of existing light-weight representations. We recognise the importance of the iStar community to be engaged in the definition of iStar extensions mechanisms. So, this our preliminary proposal can be a starting point for discussions about the theme and future inclusion of extension mechanism in iStar 2.0.

As future work, it is important to include a new version of the extension mechanisms and to propose a tool able to use them in a practical way. Another required important contribution is to formalise the iStar metamodel with extension mechanisms with a formal/semi-formal language such as *Alloy*. Last but not least, the proposal of a process to guide the proposal of future iStar extensions is an ongoing work.

Acknowledgements

The authors thank to CNPQ / Brazil (Conselho Nacional de Desenvolvimento Científico e Tecnológico) by the financial support to the execution of this work, Universidade Federal do Ceará, LER-UFPE and NOVA LINCS Research Laboratory (Ref. UID/CEC/04516/2013).

References

1. Amyot, D., Ghanavati, S. , Horkoff, J., Mussbacher, G., Peyton, L., Yu, E. Evaluating goal models within the goal-oriented requirement language. Int. J. of Intelligent Systems, 2010.
2. Amyot, D., Mussbacher, G. User Requirements Notation: The First Ten Years, The Next Ten Years, Journal of Software, Vol. 6, Nr. 5, 2011
3. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J. Tropos: an agent-oriented software development methodology. Auton. Agents MAS Journal, 2004.
4. Dalpiaz F., Franch X., Horkoff J. iStar 2.0 Language Guide, 2016
5. Giorgini, P., Massacci, F., Zannone, N. Security and trust requirements engineering. In: International School on Foundations of Security Analysis and Design III, 2005.
6. Gonçalves, E., Castro, J., Araújo, J., Heineck, T. A Systematic Literature Review of iStar extensions. The Journal of Systems and Software, v. 137, p. 1-33, 2018.
7. Gonçalves, E., De Oliveira, M., Monteiro, I., Castro, J., Araújo, J. Understanding what is important in iStar extension proposals: the viewpoint of researchers, REJ (Under review).
8. ITU-T Z.150, Formal description techniques (FDT) – User Requirements Notation (URN), available in https://www.itu.int/rec/
9. Kitchenham B., Pfleeger S. Principles of Survey Research, Soft. Engineering Notes, 2002.
10. OMG, Unified Modeling Language specification 2.5.1, available in https://www.omg.org/
11. Yu, E. Modelling Strategic Relationships for Process Reengineering. U. of Toronto, 1995.