i*-REST: Light-Weight *i** Modeling with RESTful Web Services

Zinayida Petrushyna, Alexander Ruppert, Ralf Klamma,
Dominik Renzel, and Matthias Jarke

RWTH Aachen University,
Information Systems and Databases Chair
Ahornstr. 55, 52056 Aachen, Germany
{petrushyna,ruppert,klamma,renzel,jarke}@dbis.rwth-aachen.de

Abstract. Environmental changes drive the evolution of organizational software systems. *i** allows to model continuous requirements on social and intentional aspects of organizational software systems. However, partially computer-enacted workflows are not well supported by current tools. In this paper, we propose i*-REST, an Open Source collection of RESTful Web services that are designed to manipulate existing *i** models on-the-large. More specific, we describe two services in detail. The first one allows the creation, modification and storage of Strategic Dependency models. The second service embeds *i** model visualizations in arbitrary Web pages. We have evaluated the services in a case study on learning forums. We introduced a workflow for the refinement of existing archetypical *i** models of learning communities without human intervention. Maintenance and evolution of existing *i** models will be much easier, in particular when more i*-REST services become available.

Key words: RESTful Web service, *i** modeling

1 Introduction

Organizations evolve as they require rapid adaptations to environmental changes. Sustainable success of organizations is only possible by continuous support of their systems [7]. Early requirements of an organizational system aim to support engineers by highlighting system's environment, its players and customers. Then the engineers can model new systems, extend existing systems or refine them.

Jarke et al. [7] use *i** for continuous support of evolving organizations and communities. The same research group extended *i** by adding evolution links between roles that actors acquire in the time of system evolution [10]. Business process re-engineering is one of the possible *i** applications [4]. A technical contribution that allows to automatically create *i** models is needed to keep *i** models consistent after changes in organizations or communities.

J-PRIM [5] focuses on prescriptive construction of *i** models that utilize Detailed Interaction Script. The tool expects users to assist by data gathering. Many other *i** tools allows to create *i** models only with user interaction.

In this paper, we describe RESTful services that allow to maintain $i*$ models by receiving commands in the form of REST requests from other applications. i*-REST provides a storage with versioning support where models are stored in iStarML [2] format. The visualization of models is represented in Scalable Vector Graphics (SVG), which can be embedded in any Web page. Last but not least, the i*-REST services are implemented as part of an Open Source peer-to-peer system. The services provide the basis for the algorithmic refinement of $i*$ models without human intervention. We present the example of the refinement in our case study.

In the following, we give an overview of the i*-REST service architecture. After that, we discuss creation, editing and deleting models using REST requests. Later we point out the service possibilities to visualize, import and export $i*$ models. We finish the paper with our case study, discussions and plans for the future.

2 The i*-REST Infrastructure

i*-REST consists of two services. One of the services maintains $i*$ models by creating, editing, deleting and storing them as iStarML files. The other service creates SVG files that can be viewed by any Web browser. The architecture of i*-REST (Figure 1) allows the services to be distributed in peer-to-peer systems.

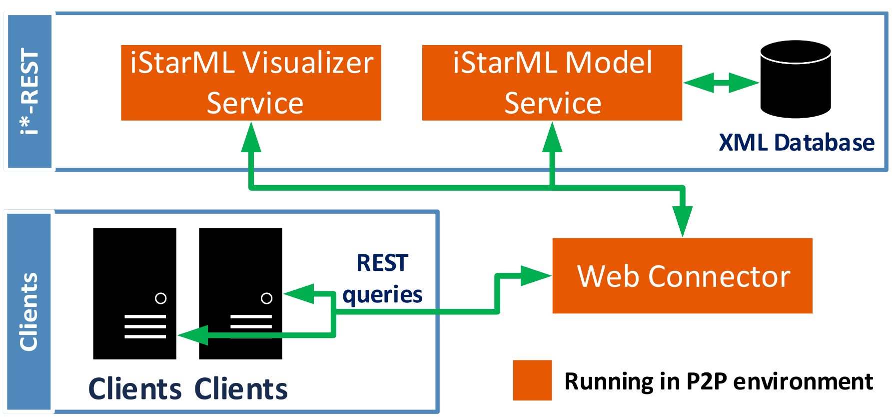

Fig. 1: Overview of the i*-REST architecture.

The Web connector handles REST requests from outside and converts them into method calls of the i*-REST services.

The XML database serves as an $i*$ model repository. An eXist[1] database allows the storage of XML files and their versions efficiently.

[1] http://exist-db.org/exist/apps/homepage/index.html

The *iStarML Model Service* creates, modifies and retrieves i^* strategic dependency models. An example of an iStarML file is given in Listing 1.

Listing 1: Initial model as an iStarML file

```
<istarml>
  <diagram name="diagram">
    <actor id="1" name="Homework" type="actor"/>
    <actor id="2" name="Teacher" type="actor"/>
  </diagram>
</istarml>
```

The RESTful API of i*-REST allows to create and modify i^* models using requests that are related to the iStarML syntax. The first request (Table 1) results in the code from Listing 1. The goal dependency between both actors is created by requests 2-4 and will add the lines from Listing 2. These dependency links and the goal dependency are deleted by the request 5. For the complete list of supported REST requests an interested reader can visit the i^* wiki page of i*-REST[2].

Listing 2: An intentional element, its depender and dependee in iStarML format

```
        <ielement id="3" name="evaluate" type="goal">
            <dependency>
                <depender aref="1"/>
                <dependee aref="2"/>
            </dependency>
        </ielement>
```

The service keeps iStarML files always valid by rejecting invalid operations, e.g. adding an actor link to an intentional element is not possible. Moreover, the service handles changes in models intelligently, e.g. a node is deleted with all dependency links or an intentional element is deleted together with its dependency links.

The *iStarML Visualization Service* creates visual representations of i^* models by converting iStarML files into SVGs that can be embedded in arbitrary Web pages. The graph representation of a model is generated using the yFiles[3] library. It ensures a compact graph layouting and usage of colors for nodes and labels.

We create a Web page that allows user interactions with a model repository. The search functionality is realized by utilizing XQuery [11]. A user can search for the names of models, actors and intentional elements.

The visualization representation can be navigated, similar to an online map, by dragging and zooming with a mouse or a keyboard. Comments specified inside an iStarML file are shown as tooltips on top of the SVG representation. A user can download a visualized model as a SVG file (export) and import a local iStarML file to store and visualize it.

[2] http://istar.rwth-aachen.de/tiki-index.php?page=i%2A-REST

[3] http://www.yworks.com/en/index.html

Request	Description
`GET Collection/model1`	Returns the model stored in a model repository.
`PUT Collection/model1/ielement /3?type=goal&name=evaluate`	Creates an intentional element of type goal (ID=3) in the model.
`PUT Collection/model1/ielement /3/depender/1`	Adds a dependency link between the actor (ID=1) as a depender and the intentional element (ID=3) in the model.
`PUT Collection/model1/ielement /3/dependee/2`	Adds a dependency link between the actor (ID=2) as a dependee and the intentional element (ID=3) in the model.
`DELETE Collection/model1/ielement/3`	Deletes the intentional element (ID=3) in the model.

Table 1: Examples of REST requests.

3 Case Study

Similar to organizations, learning communities have their actors, policies, rules and tools that they have to adopt because of environmental changes. We model learning communities using *i**, where learning community environments and their customers are *i** actors and learning materials, goals, tools, and tasks are dependencies between the actors. We present archetypical community models in learning environments like the *innovative community*, the *question-answer community*, and the *expert community* [9].

In our study we investigate two learning forums: one with 430K posts and 21K users over 9 years, the other with 200K posts and 25K users for more than 13 years. Both are learning forums: the first prepares language learners for tests; the second supports physicians in lifelong learning. We detect communities using the Louvan algorithm [1], analyze communities applying social network analysis and detect self-regulated learning patterns [8]. Moreover, we define a number of roles community members may play, e.g. *answering persons*, *questioners*, *trolls*, *conversationalists*, and many others. We extract non-human agents using Natural Language Processing tools. Based on our findings, we create *i** models for communities using i*-REST.

A community model suits one of the archetypical *i** model, e.g. *innovative* or *expert* community models. After finding a suitable model for the community we mine community goals [8]. We refine the model according to the goals automatically as soon as goals change and thus community requirements change. We can embed the *i** model into community Web pages for an immediate view. We hypothesize that community users, managers or stakeholders get an opportunity to find missing resources or actors that may fulfill community goals. For example, if community users *want to succeed* in the language learning test, a community has to invite community members that take roles of answering persons and experts who have already succeeded in the test. Moreover, the provision of learning materials is considered helpful.

Before we implemented the services of i*-REST in the community, we evaluated their usability. Seven developers simultaneously used the i*-REST services for creating and editing small models (<10 actors, <10 dependencies). Each participant worked with her own set of models. We created a Web page where users sent REST requests to create and modify *i** models. The idea to use REST requests was well accepted, the generated visualizations and service response times were satisfactory and the developers preferred Web pages to native clients for *i** model visualization. However, the last result should be precisely investigated, e.g. by comparing Web browser-based and desktop tools for *i** modeling.

4 Discussions and Future Work

In this paper we introduce the light-weight technical contribution. The purpose of the contribution is the computer-enacted support of continuous requirements changes. Our implementation allows to maintain and store *i** models and their storage using RESTful services. In our case study, we realize the workflow of modeling continuous requirements without human intervention. We use i*-REST for creating models based on continuous analysis of learning communities appearing in Web forums. Model visualization of Web-based learning communities are helpful for their stakeholders. They can analyze community situations, adopt communities to suit a successful pattern, and provide additional functionalities to support communities to achieve their goals.

On the example of learning forums' environment we introduce continuous support of evolving communities. Applying i*REST services that analyze forum communities we keep *i** models of communities consistent with their evolution. Stakeholders can learn from existing *i** models and choose design alternatives [6] to refine community environments.

We will extend i*-REST by supporting Strategic Rationale models, visualizing differences between model versions, adding evolution links between user roles [10] and enabling collaborative modeling [3].

The RESTful approach provides a strong basis for interoperability and flexibility, where services and users can harness the full power of a growing service repository. Introducing i*-REST, we encourage the *i** research community to contribute with other *i** RESTful Web services. They may facilitate collaborative modeling and sharing of i* models between organizations and communities.

5 Acknowledgment

This work was supported by NRW State within the B-IT Research School and RWTH Aachen University.

References

1. V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech. (*, 2008.
2. C. Cares, X. Franch, A. Perini, and A. Susi. Towards Interoperability of i* Models Using iStarML. *Comput. Stand. Interfaces*, 33(1):69–79, 2011.
3. M. Derntl, S. Erdtmann, P. Nicolaescu, R. Klamma, and M. Jarke. Echtzeitmetamodellierung im Web-Browser. In K. D. HG and U. Reimer, editors, *Modellierung 2014. Lecture Notes in Informatics*, volume 225, pages 65–80, DE, 2014. Gesellschaft für Informatik e.V.
4. Eric Siu-Kwong Yu. *Modelling strategic relationships for process reengineering.* PhD thesis, University of Toronto, Toronto and Ont. and Canada and Canada, 1995.
5. G. Grau, X. Franch, and N. A. Maiden. A Goal-Based Round-Trip Method for System Development. In *In Proceedings of the 11th International Conference on Requirements Engineering: Foundations for Software Quality (REFSQ'05) Key: citeulike:8774851.*
6. A. Hilts and E. Yu. Intentional Modeling of Social Media Design Knowledge for Government-Citizen Communication. In M. Atzmueller, A. Hotho, M. Strohmaier, and A. Chin, editors, *Analysis of Social Media and Ubiquitous Data*, volume 6904 of *Lecture Notes in Computer Science*, pages 20–36. Springer Berlin Heidelberg, 2011.
7. M. Jarke, R. Klamma, Gerhard Lakemeyer, and D. Schmitz. Continuous, Requirements-Driven Support for Organizations, Networks, and Communities. In *Proceedings of the 3rd International i* Workshop, Recife, Brazil, February 11-12, 2008*, pages 47–50, 2008.
8. J. A. Krenge, Z. Petrushyna, M. Kravcik, and R. Klamma. Identification of Learning Goals in Forum-based Communities. In *Proceedings of the IEEE 11th International Conference on Advanced Learning Technologies*, pages 307–309, Athens and Georgia and USA, 2011. IEEE Computer Society.
9. Z. Petrushyna, R. Klamma, and M. Kravcik. Designing During Use: Modeling of Communities of Practice. In *IEEE International Conference on Digital Ecosystems and Technologies*, 2010.
10. A. Roesli, D. Schmitz, G. Lakemeyer, and M. Jarke. Modelling Actor Evolution in Agent-Based Simulations. In G. Vouros, A. Artikis, K. Stathis, and J. Pitt, editors, *Organized Adaption in Multi-Agent Systems*, volume 5368 of *Lecture Notes in Computer Science*, pages 126–144. Springer Berlin Heidelberg, 2009.
11. P. Walmsley. *XQuery.* O'Reilly, Farnham and Calif, 2007.