Modelling the Aggregation of Multimedia Data to Connect the Inputs and Outcomes of a Variety of Tools

Emilio Julio Lorenzo, Maria Felisa Verdejo
LTCS Group, Dpto. Lenguajes y Sistemas Informáticos
UNED University
Madrid, Spain
{emiliojulio, felisa}@lsi.uned.es

Abstract—This paper presents an approach for helping developers to build or integrate educational tools into virtual learning communities, where they have to interact with other tools. Our proposal uses a declarative language based on Web forms. Thus, the products that tools create (outcomes), or need for configuration (incomes) can be modeled. A study about different integration scenarios and the added values of using this approach are showed, including a real testing scenario.

Keywords—metamodel; learning tool; interoperability

I. INTRODUCTION

Currently learning activities are carried out in social structures supported by a technological infrastructure. In these scenarios, students and teachers work and communicate using a variety of tools [1]. Along the learning process, outcomes are generated in an incremental process, so that the outputs of one step became inputs for the next one. How to deal with data and process interoperability is an important issue nowadays, furthermore in educational settings, where usually there is not expertise available to program ad-hoc solutions.

The main objectives of our research are: Firstly, integrate external tools into the technological infrastructure supporting a virtual learning community. In our case it is supposed to include, at least, a Learning Object Repository (LOR). Secondly, provide a protocol and technical recommendations to help developers with the integration of their tools. That means non-intrusive code and in a declarative way when possible. Thirdly, make a study about integration scenarios for both external tools and new ones in order to incorporate them into the community environment. And fourthly, test the proposal in a real scenario and study the results.

There is a variety of languages to help developers to model learning activities and integrate their tools into a virtual community. The Education Modeling Languages (EML) proposals describe activities and learning scenarios [3] as well as different ways to package educational content, like [4]. However, not many approaches help developers to implement them into their tools. Regarding content authoring tools, they usually deal with hypermedia or visual representations [5], but do not consider how to integrate other tool products as learning objects. Moreover, other approaches contain their own architecture and to include external tools in this case is a complex task [6].

The chosen solution to reach our objectives listed above is to use Cards-Metamodel [7], that offers a metamodel engine acting like a modeling language to wrap tool outcomes in a flexible way. Its main feature, and the reason of its selection, consists of the model is built in a declarative way and with a typical HTML form interface. This feature covers the objectives of a flexible and easy-to-use solution. The structure of the rest of the paper is as follows: section II describes a real testing scenario. Section III offers the chosen solution. In sections IV and V we propose several classifications and added values related with this research. Section VI shows a proof of concept. The final section offers a summary and future work.

II. THE TESTING SCENARIO

In this section, a real tool integration scenario to clarify the proposal is introduced. The scenario was the development of a mobile application to draw the path that K-12 students, along with teachers and guides, walk in a field trip. This tool was named PDAMaps as codename. In this kind of activities there is sometimes a lack of internet connection, so a non-online application is needed. The infrastructure for the virtual community environment included a repository and other tools used by students and teachers to generate learning objects of different type of content.

A specialized programmer in mobile and GPS software developed the tool. But it was necessary to configure the tool by setting an image representing the map and a pair of coordinates (top left, bottom right) in order to find the current location in the map, by interpolation techniques. Although the tool was stand-alone, its configuration must be carried out by teachers on internet using a web page. Therefore, an authoring system to configure the tool in order to carry on an activity was necessary. Another problem was how to find the maps. We wanted all possible reusable resources to be learning objects, and not only digital files. Thus, the PDAMaps had to be able to manage the maps (and their associated coordinates) as learning objects in its authoring system. Finally, the tool outcomes had to be also stored as learning objects, including metadata related to their use in a learning activity.
III. THE PROPOSED SOLUTION

Cards-Metamodel is proposed to deal with the previous problems of integration and configuration of tools. This approach allows creating a description of the artifacts handled in any educational tool, as well as to organize, pack, share and store these objects. It is based on creating models as HTML forms in a declarative way [7].

Fig. 1 shows an example of design and implementation phases of a model with the Cards-Metamodel. The tool products are represented as HTML forms (Card). Cards-Metamodel is used to define those forms by declaring a model (Card Template) acting as template for the final form. Therefore, a declarative model language based on sections and qualified fields, arises.

In the testing scenario, the products that the PDAMaps authoring system needs are an image file for the map and a text for the coordinates. Including the coordinates in the image file as gif metadata has a high coupling with the file format and makes it difficult to show or modify them. Using Cards-Metamodel, a template is designed with several fields (see Fig 2): One section for the image, with several field types which allows several sources, including learning objects from the repository. Other section contains two string fields for coordinates. The last section offers information about the location, what was marked as metadata. Once the template is finished, instructional designers use the Cards-Metamodel web interface to instantiate the template. An example is showed in Fig. 2. The image of the map, stored in the field named “ImageURL”(1), is collected from a LOR as a Learning Object (2) what contains the image (3). That object was in turn generated in a previous learning activity by another tool. Next, PDAMaps can get the card with the map and extract the data to configure itself.

Along with Cards-Metamodel tool, our proposal includes a Java library with a high and low abstraction level API. Developers should know it to manage the models of their tools. The authoring web interface has been implemented using standard Java frameworks and following the model-view-control pattern. Thus, modifications are easy to make if necessary. An implementation protocol is proposed to improve the interaction between external tools and Cards-Metamodel consisting of the following:

1. Create, with Cards-Metamodel web interface, the model, i.e. Card Template.
2. Mark content fields which are also metadata, in order to build a learning object.
3. Code with the Cards-Metamodel library. This means, creating an ordinary map data structure, linking products (like digital resources) with card fields. Internal tool context data, like social context, can also be included as metadata.
4. Create the Learning Object from the previous card.
5. Send the Learning Object to the repository by Web Services.

IV. TOOL INTERACTION SCENARIOS

In the following we propose several integration scenarios allowed using our proposal.

A tool can interact with Cards-Metamodel in the following ways:

- Using the library. To build card templates and cards. The result format can be from a simple XML file to a more complex package similar to SCORM standard to aggregate digital content. The communication with repository is also offered in an easy programming way, hiding web services APIs.
- Using the authoring web interface. By using URL links that return to the tool when the process is finished. This method is complementary to the previous one. E.g. once the template or card is created via web interface the result could be managed by the tool, using the java library.

Depending on the products to be wrapped:

- Configuration resources. It concerns the artifacts that the tool needs to configure and prepare an activity or task. E.g. the testing proposal.
- Final result. That is, the tool outcomes, when it is used in an activity, supporting prosumer users. As in the previous definition, this result is, usually, one or several digital resources in different formats. An example is showed in [7].

Depending on the state of the tool implementation:

- Wrapping. When the application has already been developed, usually based on plain java objects. Here, a template is designed to reify and wrap the existing model.
- Ascribing. When the application implements its own model (internal or final) using Cards-Metamodel language.

Figure 2. Modeling PDAMap configuration with Cards-Metamodel
V. ADDED VALUES

Our proposal offers several added values to develop educational tools that are not always present in other approaches. The main contributions are:

**Authoring.** It has already discussed how content can be created using the Cards-Metamodel web interface, i.e., creating a card and viewing the result with the same interface later. These functionalities can help developers to easily build authoring tools.

**Adding semantic information.** The main contribution to the semantics is that we can design the card, which contains the products, with all the extra fields we need to include metadata information. E.g., if the product is an image, an extra text field with a description will help as metadata for searching. Pedagogical products can also be organized in logical blocks, even if they are not linked conceptually. The name of the field can also be designed as a name space, allowing more complex structures, for instance, to support standards like SCORM Content Aggregation. E.g., if the product is a course, instead of naming a field “course name”, it will be named “organization:Course” and later is transformed into xml format, like <organization><course><title>. The content of a field is constrained by a concrete type that assures its syntax correctness. Regarding filling metadata is an issue that authors or students not always carry out completely. Cards-Metamodel can automatically populate metadata values from the content of a card [2].

**Reusing models.** A new card template can be built from existing ones, so that authors can use previous work and avoid the starting “from scratch” authoring problem. In that way, collaborative authoring is supported, because previous cards templates can be refined. To organize and make the search of templates easier, they can be classified in several ways, like activity or domain.

**Internationalization.** Each field and vocabulary values can be written, and showed in the languages prescribed by the author of the template. For instance, if the pedagogical content author defines a field in English, Spanish and Finish, the students of each country will see this field in their own language. This allows carrying out activities between students of different nationalities.

**Tool interoperability.** Several tools can interoperate into a networking infrastructure to create complex activities. In this scenario Cards-Metamodel acts as an ‘interlingua’ among tools to share and interpret their results [7]. This functionality gives uniformity and ‘corporate vision’. Fig.3 shows an example of tools interoperability and how the tool’s outcomes could be the incomes of another one or how to combine several outcomes to create a unique object.

VI. TESTING THE PROPOSAL

Because of the declarative way to create the model, as a web page form, the developer did not need to learn any kind of modeling language. Few lines of code were used and following the steps implementation protocol explained previously. The authoring system was fully supported by the web interface, avoiding further work and new web page developers. The amount of time estimated for implementing the authoring system and the integration of PDAMaps into our educational environment was estimated to be over two months. This integration consisted of creating learning objects with metadata and communication with a repository via web services. Using our proposal, the time was reduced to two days, training included. Finally, asking four instructional designers, they preferred a common interface to configure the tools instead of different ones. Therefore, our proposal would be the first step to a further tool integration model, including instructional processes.

VII. CONCLUSIONS AND FUTURE WORK

A proposal to help developers to integrate their tools into a virtual learning community has been described. A declarative model language in web form fashion, named Cards-Metamodel, is used. A study of how programmers can use this solution in different scenarios is also described. Finally, the result of the proof of concept suggests that the programmer agrees with our approach to integrate external tools. Besides, he saves a considerable amount of time developing the tool.

The next step in our research is to go further with the semantics of the fields and integrate tools using other topics in e-learning, like the social or user model.

ACKNOWLEDGMENT

We thank MICINN for the support of Projects TIN-2009-14317-C03-03 and TIN2004-04232 and Region of Madrid for Network S2009-TIC-1650

REFERENCES


