



HAL
open science

Language Independent Rules for Suggesting and Formalizing Observed Uses in a Pedagogical Reengineering Context

Vincent Barré, Christophe Choquet

► **To cite this version:**

Vincent Barré, Christophe Choquet. Language Independent Rules for Suggesting and Formalizing Observed Uses in a Pedagogical Reengineering Context. IEEE International Conference on Advanced Learning Technologies (ICALT'2005), 2005, Kaohsiung, Taiwan. pp.550–554, 10.1109/ICALT.2005.189 . hal-01452355

HAL Id: hal-01452355

<https://hal.science/hal-01452355>

Submitted on 13 Mar 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Language Independent Rules for Suggesting and Formalizing Observed Uses in a Pedagogical Reengineering Context

Vincent Barré, Christophe Choquet
LIUM / IUT de Laval
52, rue des docteurs Calmette et Guérin
53020 Laval Cedex 9, FRANCE
{Vincent.Barre, Christophe.Choquet}@univ-lemans.fr

Abstract

The process we propose in this paper is in keeping with a more general thought about e-learning systems reengineering. It deals more precisely with pedagogical scenario reengineering by assisting designers in the uses tracking definition phase in order to allow them to a posteriori understanding of the learning session sequence. From this point of view, our process presents some advantages. First, it allows the recommendation of uses to track from the designer built scenario structure. Then, it proposes a formal way to transcribe these observed uses in a design interpretable language, and thus to compare a priori scenario to the one resulting from a posteriori observed uses. Lastly, it relies on an analysis at a meta-model level of the pedagogical modeling language chosen by designers and, therefore, allows complete dissociation from a chosen language.

1. Introduction

The desynchronization of the two major teaching roles – course design and tutoring – in distance education, penalizes iterative optimization of the system quality by not taking into account uses with a reengineering objective. That's why in [2] we have proposed an extension of IEEE-LTSA (Learning Technology Systems Architecture) meta-architecture model [5]. This proposition explicitly integrates a step dealing with the observation and component analysis of distance learning systems and the learning process actors in an iterative process, guided by design intentions. We underline in particular, the need for a formal description of the design point of view of the scenario, called *prescriptive scenario*, as well as assistance in uses analysis by comparing *descriptive scenarios* (an *a posteriori* scenario that effectively describes the learning situation's sequence [7]) with the predictive scenario. This produces information,

significant for designers from a pedagogical point of view, when they perform a retro-conception or a reengineering [4] of their systems.

The designer role in such a process is then, not only to establish the predictive scenario of a given learning situation, but also to anticipate descriptive scenario construction by defining situation observation needs which allow the effective evaluation of the learners' activity. In the framework of REDiM project (Reengineering Driven by Models), we are more particularly interested in supporting the implementation of those two tasks in a context where designer uses an activity-centered pedagogical modeling language, from the Educational Modeling Languages (EML) family [8].

Although there is a current international willingness to result in a normalized EML, many languages coexist [10], even if *Learning Design* from the IMS consortium (IMS/LD [9]) seems close to becoming standard. This is why all of our work, although exemplified with IMS/LD, wants to be independent from designer modeling languages.

We propose in this paper a rule set, reasoning about IMS/LD meta-model (that is, its XML-Schema) and whose inferences on the predictive scenario allow us to define a set of relevant uses to track during the learning session. We moreover propose a method intended to represent descriptive scenarios which is based on an extension of the modeling language used to express the predictive scenario. This method has many advantages: (i) to prepare language extensions to use defining rule inferences, (ii) to facilitate descriptive scenario interpretation by comparing them with predictive ones, both being expressed in the same language, and (iii) to be independent from the used modeling language, as long as its meta-model is expressed with an XML-Schema, which is the case for most of them. A last part of this paper is devoted to a

use case showing the process dynamics and clarifying the introduced rules.

2. Recommendation and representation of relevant observed uses

In this part, we will present a method allowing the recommendation of uses to track in order to compare predictive and descriptive scenarios. The original feature of our approach is that recommendations are made considering the predictive scenarios language meta-model, rather than an analysis of scenarios by themselves or an analysis of the structure of observed uses supplied by the LMS.

We will limit our approach to educational modeling languages expressed by XML-Schemas, for which recommendations are deduced from their elements composition. More precisely, in an XML-Schema, two types of elements can be defined: those of *complexType* (that can contain sub-elements and/or can be qualified by attributes) and those of *simple* type. In order to highlight observed uses linked to elements of any EML (IMS/LD for example), those two sorts of elements bring complementary information: observed uses linked to *complexType* elements will be defined by production rules (with regard to the meta-model and what we can expect to find in tracks for those elements), whereas those linked to *simpleType* are just extracted from computer tracks. But, whatever element type, rules used to produce relevant observed uses, will not rely on the language described by the schema (that is, on the adopted EML).

This rule set falls within the operationalization process of a pedagogical scenario reengineering, as tackled in [6] and brings operational answers to ideas on pedagogical scenario lifecycle [7]. However, the rule set presented here is a first set which must be enriched afterwards.

The process that we propose here will consist in “enriching” the predictive scenario with the designer’s required observed uses. A first set of rules (numbered 1 to 3 below) brings enrichment by synthetic information connected with observed routes. A second set (numbered 4 to 6) draws the designer’s attention to the meaningful configurations from a reengineering point of view. Finally, although our rules are not conceived for that, it will be interesting to be able to produce observed uses allowing the representation of individual learner routes.

Rule 1: For any *complexType* element, it is interesting to focus on the total number of occurrences of that element found in computer tracks.

Rule 2: When an element is of *simple* type (for example, *integer*, *string*...), it is interesting to focus on the value(s) of this element (taking into account its different values, if applicable).

Rule 3: When an element references or contains another element for which we have determined observed uses, we import those observed uses into element context. This implies not only that all observed uses linked to the included or referenced element are accessible, but also that it is possible to obtain a restricted vision of the current element.

Rule 4: For an element whose *complexType* contains a choice between several elements, it is interesting to focus on elements effectively found in computer tracks.

Rule 5: For an element whose *complexType* contains an element sequence with an unbounded *maxOccurs* property, it is interesting to examine the descriptive scenario so as to locate unused elements.

Rule 6: For an element whose *complexType* is empty, there is no corresponding observable.

These rules allow uses tracking recommendations from the XML-Schema structural analysis of any designer language; we will now consider how to include those observed uses in the same language. More precisely, we will try to produce an “enriched” scenario with information coming from these observed uses as in [1]. In order to do that, main idea is to begin with a predictive scenario, containing information on the predictive session sequence, and to append information mined from tracks on designer queries. In concrete terms, for each rule, obtained observed uses will be appended to the rule’s target element definition (in order to enrich the predictive scenario). This will happen through the designer’s language XML-Schema transformation, in order to incorporate our extensions. We underline that this new XML-Schema will constitute an extension of the original one, and, therefore, the predictive scenario will also be a valid document for our extended schema.

3. Use case

Over 3 years, we have tested a learning unit dedicated to improving learners' HTTP Server programming skills with about fifty French students [3]. This learning unit was composed of the following seven resources: (1) *introduction* (a video), (2) *text study* (some reference texts), (3) *slide show*, (4) *case study* (some exercises), (5) *check up* (multiple choice questionnaires), (6) *learning by doing* (where learners use acquired knowledge in order to build, step by step, a Java Web server) and (7) *using produced system* (collective activity about Web site conception using a previously produced Web server). In our experiment, we have established a predicted scenario but we choose to keep learners free to navigate as they want to between the six first activities. After each learning session, collected computer tracks are analyzed and compared with the predicted scenario, in order to improve the proposed resources.

In this use case, we will focus on observed uses linked to properties that are part of the IMS/LD level B specification. These properties provide a mechanism allowing the storage of information about learners, learners groups, ... they also allow scenario individualization based on learner profiles or previously obtained scores (in this learning unit or in another). They are mainly used and updated in the *method* section (representing the scenario dynamics) of a *Learning Design*, in two ways: (1) when a learner completes a scenario element (activity, *act*, *play*...) and/or (2) in a conditional rule (if a given condition arises, then a given property is to be updated).

In our experiment, properties are used to transcribe scores obtained by learners in multiple choice questionnaires:

```
<imsld:learning-activity identifier="LA5-AutoEval-QCM1">
  <imsld:title>S'auto-évaluer (QCM1)</imsld:title>
  <imsld:environment-ref ref="LO3-QCM1.html" />
  <imsld:activity-description>
    Tester ses connaissances sur les protocoles du Web
  </imsld:activity-description>
  <imsld:complete-activity>
    <imsld:user-choice />
  </imsld:complete-activity>
  <imsld:on-completion>
    <imsld:change-property-value property-ref="P-QCM1-score"
      property-value="score"/>
  </imsld:on-completion>
</imsld:learning-activity>
```

This activity is associated to *R-learner* learners with a *role-part* element:

```
<imsld:role-part identifier="Rp1-learner">
  <imsld:role-ref ref="R-learner" />
  <imsld:learning-activity-ref ref="LA5-AutoEval-QCM1" />
</imsld:role-part>
```

When examining computer tracks obtained after a learning session, it is possible to extract the value taken by *score* when it was assigned. However, this

information is not very useful as such. It will be more informative if it can be contextualized, for example associated with the corresponding questionnaire, or the position of the questionnaire compared with other activities. So, let us examine what we can do with our rules.

To begin with, our first rule allows the transcription of the number of questionnaire occurrences found in the computer tracks. In order to represent this information, we have to modify the *learning-activityType* element definition in IMS/LD XML-Schema :

```
<xs:complexType name="learning-activityType">
  <xs:sequence>
    <xs:element ref="title" minOccurs="0"/>
    <xs:element ref="learning-objectives" minOccurs="0"/>
    <xs:element ref="prerequisites" minOccurs="0"/>
    <xs:element ref="environment-ref" minOccurs="0"
      maxOccurs="unbounded"/>
    <xs:element ref="activity-description"/>
    <xs:element ref="complete-activity" minOccurs="0"/>
    <xs:element ref="on-completion" minOccurs="0"/>
    <xs:element ref="metadata" minOccurs="0"/>
  </xs:sequence>
  <xs:attributeGroup ref="attr.identifier.req"/>
  <xs:attributeGroup ref="attr.isvisible"/>
  <xs:attributeGroup ref="attr.parameters"/>
  <xs:attribute name="observedOccurs"
    type="xs:nonNegativeInteger"/>
</xs:complexType>
```

The *LA5-AutoEval-QCM1* activity can be described in the following way in the descriptive scenario:

```
<imsld:learning-activity identifier="LA5-AutoEval-QCM1"
  observedOccurs="10">
  <imsld:title>S'auto-évaluer (QCM1)</imsld:title>
  . . .
</imsld:learning-activity>
```

Next, the second rule transcribes the *P-QCM1-score* property values. Compared with the tracks' immediate exploitation, the main advantage here is to transcribe successive values taken by a property (for example, the multiple choice questionnaire can be first used in order to evaluate knowledge to be acquired, and then, in order to evaluate that effectively acquired). So, we want to extend the *initial-value* element definition :

```
<xs:element name="initial-value" type="xs:string"/>
```

so as to transcribe successive values taken by this property. In order to do that, we will begin creating a new *complexType* describing this element:

```
<xs:element name="initial-value" type="initial-valueType"/>
```

the *initial-valueType* type being defined as an extension of the *xs:string* type :

```

<xs:complexType name="initial-valueType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="history" use="optional">
        <xs:simpleType>
          <list itemType="xs:anySimpleType"/>
        </xs:simpleType>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

```

Let us then examine the *change-property-value* element and, particularly, the *property-value* element that it makes use of:

```

<xs:complexType name="property-valueType" mixed="true">
  <xs:annotation>
    <xs:documentation xml:lang="en">The element property-value should contain either text, or langstring, or calculate, or property-ref.</xs:documentation>
  </xs:annotation>
  <xs:choice minOccurs="0">
    <xs:element ref="langstring"/>
    <xs:element ref="calculate"/>
    <xs:element ref="property-ref"/>
  </xs:choice>
</xs:complexType>
<xs:element name="langstring" type="xs:string"/>
<xs:element name="property-ref">
  <xs:complexType>
    <xs:attributeGroup ref="attr.ref.req"/>
  </xs:complexType>
</xs:element>

```

This element could contain either an *xs:string* element, a property reference, or “result” of a calculus. For this last possibility, an analysis of the *calculate* element shows that it is a recursive definition leading to either a property element reference, a *property-value* element, or a new expression (that is, a sub-expression). Whatever the case may be, our method allows the recommendation of some trackable uses :

- If the *property-valueType* corresponds to a string (simple type), then the second rule recommends property value observation, and eventually, transcription of its successive values.
- If the *property-valueType* correspond to a property reference, then the third rule recommends associating observed uses linked to the referenced property and contextualising this information (which is not so useful here, since only the property values seem to be interesting).
- Finally, if the *property-valueType* corresponds to the previously mentioned recursive definition, then, in all three cases (property element reference, *property-value* element and new expression), our rules recommend some traceable uses.

We can then apply our third rule to import observed uses linked to a property where the element activates the modification. At first, this allows the importation of observed uses linked to the *P-QCMI*-

score property in the *LA5-AutoEval-QCMI* learning activity context and then, in the context of the *role-part* linking this activity to the *R-learner* role.

Imported observed uses are linked to included or referenced elements, therefore there is no need to provide for them in the new XML-Schema. It is nevertheless important to represent the contextualized information. This can be done in two different ways, depending on whether the element includes or references another one. In case of an inclusion, the element description needs enriching, whereas in case of a reference we need to enrich the definition of reference containing the element. Nevertheless, whatever the case may be, definition enrichment will be done in a similar way and the same attribute can be used. For example:

```

<imsld:role-part identifier="RP30-learner">
  <imsld:role-ref ref="R-learner" observedOccurs="11"/>
  <imsld:learning-activity-ref ref="LA5-AutoEval-QCMI"
    observedOccurs="12"/>
</imsld:role-part>

```

indicates that the *LA5-AutoEval-QCMI* activity appears 12 times in the *RP30-learner* element context, and 11 occurrences of the *R-learner* role have been associated to this *role-part*. This data can later be compared to total number of learners in the *R-learner* role and the number of instances of the *LA5-AutoEval-QCMI* activity.

Let us now mention the interest of our fourth rule. In our experiment, the *LA5-AutoEval-QCMI* activity is in fact one of three multiple choice questionnaires proposed to learners. In order to represent the total number of occurrences of each questionnaire, we just extend the *activity-structureType* element definition as for the observables linked to first rule:

```

<imsld:activity-structure identifier="AS5-AutoEval"
  structure-type="selection" number-to-select="1">
  <imsld:title>Auto-évaluation</imsld:title>
  <imsld:learning-activity-ref ref="LA5-AutoEval-QCMI"
    observedOccurs="7"/>
  <imsld:learning-activity-ref ref="LA5-AutoEval-QC2"
    observedOccurs="0"/>
  <imsld:learning-activity-ref ref="LA5-AutoEval-QC3"
    observedOccurs="3"/>
</imsld:activity-structure>

```

Here, the fifth rule draws the designer’s attention to the fact that the *LA5-AutoEval-QC2* activity has been done by none of the learners during the learning session.

4. Conclusion

The process that we propose in this paper presents many advantages. First, it uses tracking recommendation from the designer’s built scenario structure. Then, it suggests a way to transcribe these observed uses, in a design interpretable language, so

as to compare predictive scenarios with the descriptive ones resulting from the observed uses. Lastly, this process relies on an analysis of the EML chosen by designers at a meta-model level, and therefore, allows us to overlook the actual language used.

Moreover, notice that the rules we are presenting here identify observation advisability, which necessarily brings the designer to consider the real need for observation. This contributes, in our minds, to a better integration of e-learning systems lifecycle actors, and to a relationship between designer prescribed observation need and the observation means, generally set up by another actor. As such, in [6] we have defined software tools that, once developed, will allow the analysis of the collected data depending both on the predictive scenario and the formal description of elements to be observed. They will produce formal representations of user comportment, based on observation needs.

5. References

- [1] Barré V., Choquet Ch., Iksal S., Corbière A., « Usage analysis in an e-learning system: LD representation significance », *IEEE International Conference on Advanced Learning Technologies ICALT'2004*, aug 30 – sept 1st, 2004, Joensuu (Finland), p. 570-574.
- [2] Corbière A., Choquet C., « Designer integration in training cycles : IEEE LTSA model adaptation », *International Conference on Computer Aided Learning in Engineering Education CALIE'04*, Grenoble (France), feb 16-18, 2004, p. 51-62.
- [3] Corbière A., Choquet C., « A model driven analysis approach for the re-engineering of e-learning systems », *ICICTE'04*, Samos (Greece), jul 1st – 3rd, 2004, p. 242-247
- [4] Chikofsky E. J., Cross II J. H., « Reverse engineering and design recovery: A taxonomy », *IEEE Software*, 1990, 7(1), p. 13-17.
- [5] IEEE P1484.1/D9, 2001-11-30 Draft Standard for Learning Technology — Learning Technology Systems Architecture (LTSA), IEEE Computer Society, 2001.
- [6] Iksal S., Barré V., Choquet C., Corbière A., « Comparing prescribed and observed for the re-engineering of e-learning systems », *IEEE Sixth International Symposium on Multimedia Software Engineering MSE*, dec. 13-15, 2004, Miami (USA).
- [7] Lejeune A., Pernin J-P., « A taxonomy for scenario-based engineering », *Cognition and Exploratory Learning in Digital Age (CELDA 2004)*, Lisboa (Portugal), dec. 2004, p.249-256
- [8] Koper R., « Modeling units of study from a pedagogical perspective: the pedagogical meta-model behind EML », Open University of the Netherlands, 06/2001.
- [9] Koper R., Olivier B., Anderson T. eds., « IMS Learning Design Information Model », IMS Global Learning Consortium, Inc., version 1.0, 20/01/2003.
- [10] Rawlings A., Rosmalen P., Koper R., Rodriguez-artacho M., Lefrere P., « Survey of Educational Modelling Languages (EMLs) », CEN/ISSS WS/LT Learning Technologies Workshop, version 1, 19/09/2002, pp 80.