Design by Contract-Based Selection and Composition of Learning Objects

Salvador Sánchez-Alonso[1], Miguel-Ángel Sicilia[1], José-Manuel López-Cobo[2], Sinuhé Arroyo [1]

[1] University of Alcalá, Computer Science Department, Madrid, Spain
{salvador.sanchez, msicilia}@uah.es
sinuhe.arroyo@alu.uah.es
[2] iSOCO, Madrid, Spain
ozelin@isoco.com

Abstract. Selection and composition of learning objects are two essential activities in automated approaches to Web-based learning. Such activities require high-quality metadata records that are not only conforming to current specifications and standards, but that provide clear system-oriented run-time semantics that support automated decision processes. In this paper, the *Design by Contract* paradigm is described as a method to formally specify and drive selection and composition of contents aimed at concrete learning requirements. In addition, an architectural mapping for such approach to Web Service technology is described, which provides a flexible integration mechanism in a context of heterogeneous and dynamic learning content-providers.

1 Introduction

Learning Management Systems (LMS) are a concrete category of *Web-Based Information Systems* aimed at delivering diverse kinds of learning experiences. A number of evolving specifications and standards for learning contents have fostered consistency in format and description of Web learning contents [1], but they still lack a level of semantic specification enough to enable consistent runtime automated semantics. This has lead to loose metadata creation practices resulting in learning content that doesn't meet the required completeness [10] and consistency [7] to serve as the basis for common automated LMS-initiated behaviors –like selection and composition. In addition, the roles of relationships are not free of ambiguity [3], which seriously hampers the possibilities of consistent composition.

Selection of learning objects requires complete enough metadata records to allow an LMS to decide for inclusion of a given object in the ongoing composition. In addition, the composition itself requires compatibility of the metadata records of the aggregate and the parts [12], so that some properties are "propagated" between them, resembling well-known properties of aggregations in object-oriented modeling. From a technical point of view, learning object repositories can be accessed through Web services in order to provide them with the essential infrastructure to be effectively reused [2] [17]. This approach provides learning objects with a number of benefits, as expanded searching capabilities, better management of usage fees, accurate access and usage statistics and so on. But before

publishing Web service-based learning object repositories, a common way of specifying what a final user can expect from a given learning object and the conditions under which it can be used is needed. Learning object *Design by Contract* [15], a notation based on the *Learning Object Metadata* specification [5] and enhanced with richer semantics, can be used for that purpose. In previous works, design by contract [9] has been applied to the description of machine-understandable learning object metadata in the form of learning object contracts. Learning object contracts essentially allow the specification of a set of preconditions (circumstances under which the object can be used) and post-conditions (learner expected outcomes) for each learning object, which can also be used to clearly specify relationships between learning objects.

In the rest of this paper, the use of learning object-contracts to drive selection and composition processes is described, focusing on the interpretation of aggregations as the main compositional relationship. In addition, a concrete, flexible architecture based on Web Services is sketched to illustrate the actual behavior of contract-based composition services. Section 2 describes learning object contracts as a content design method for learning objects. Section 3 focuses on the specifics of the aggregation relationship, and on its consequences in the process of learning object composition. In Section 4, a Web Service-based architecture is used to illustrate the behavior of contract-based composition services. Finally, conclusions and future research directions are provided in Section 5.

2 Specifying Learning Object Contracts

The concept of 'learning object' represents an attempt to enhance the design of Web-based educational contents, focusing on their reusability in diverse learning contexts [14]. The key to reusability is the provision of metadata in standardized formats for fine-grained content items. But reusability requires precisely specified metadata records, especially if "machine-understandability" is required to build software modules that automatically retrieve and combine learning objects to form higher-level units of instruction. Unfortunately, current learning object metadata specifications do not address this and other important issues [3]. In consequence, previous work [11] [12] [15] has proposed design by contract –a technique borrowed from the object-oriented paradigm– as a way of formalizing learning object metadata records. A learning object contract can be expressed as follows:

```
rlo <URI>
   require <list_of_preconditions>
   ensure <list_of_postconditions>
```

provided that both pre- and post-conditions are expressed through assertions according to a syntax like the following:

```
[level] preconditionId.element <relationallOperator> requestedValue
        postconditionId.element <relationalOperator> value [θ]
```

Where pre- and post-condition identifiers correspond to either the learner (`lrn`), the learning context (`ctx`), or the system where the learning object is due to be executed (`sys`); element maps to a metadata element (e.g. one of those defined in LOM); and θ refers to a certain degree of credibility. This level is a way to express the fact that some learning

objects may be credited to be "more appropriate" than others, due to authoritative revisions or evaluation processes (like, for example, the peer-review assessments being carried out in the MERLOT learning object repository[1]). Finally, *level* indicates the strength of the precondition (mandatory, recommended or optional). The following example uses the just described syntax to describe a metadata instance corresponding to an introductory lesson on the use of the genitive case in English. It is intended for an Italian speaking audience, and includes the time of work required to complete the lesson (*Typical Learning Time* in LOM):

```
rlo <http://.../GenitiveCaseForItalians>
   require
     ctx.language = en ;  ctx.time = 2h ; lrn.language = it
```

Regarding postconditions, learner knowledge is obviously the principal outcome of learning activities, but other products may also be considered. For example, social relationships among learners are an important issue according to sound theories of learning [8], and learning resources that foster social activities may be considered to strength those relationships. Nevertheless, most preconditions will refer to expected learning outcomes, showing absolute or relative outcomes on the learner side. Relative outcomes compare to the learner's previous knowledge –denoted by a '–1' value–, while absolute outcomes can only be noted provided that a taxonomy for the specific knowledge domain has been defined. Following our previous example, we can have a RLO ensuring that the learner's knowledge will grow (relative outcome).

```
rlo <http://.../GenitiveCaseForItalians>
  ...
  ensure
    lrn.knows(genitive_case) > lrn.knows(-1)(genitive_case)
```

Learning object contracts can be used as a machine-understandable source of information for selection and composition, since they state requirements and outcomes in a semantically interpretable way. Nevertheless, not only self-descriptions but also relationships influence selection and composition decisions. Concretely, aggregations are critical in the interpretation of composition, as described in the following section.

3 Learning Object Aggregation and its Role in Composition

Aggregation is on the very nature of reusable learning resources: most of them are composed of others. Composition implies, among other commitments that we will examine below, that the contract of a learning object that is an aggregate of others has necessarily to be affected by the contracts of its parts. Consider for example a LOM-conformant learning object whose *1.8:AggregationLevel* is equal to 3 (course). Let's imagine that it is written in Italian. As it is a composition of lower-level objects, the language of its parts will have to be Italian as well. In cases like this, the contracts of the parts must conform to what the aggregate contact states. This way, the representation of aggregation relationships in metadata records may entail dependencies on information in other objects metadata records.

An important question is whether the parts should conform to the aggregate contract or aggregate metadata value items should be inferred from its constituent parts. As a matter of

[1] http://www.merlot.org

fact, it is a two-way process. Before the aggregate "physically" exists, the "composer agent" relies on a few directions on what the target system expects from the parts. These directions are formalized into the shape of a contract, thus creating a learning object contract blueprint specification that we call *the archetype*. This is step one. Then, an automated system looks for tentative parts (candidates) in learning object repositories, by focusing the search on the needs and restrictions expressed in the archetype. As the result of this search, a list of candidate learning objects is made up. The best candidate in the list –according to selected criteria– will be the one that will integrate the final aggregate.

Another important issue to consider is whether information on aggregation should be set in a learning object metadata record or not. Reusability asks for the components of an aggregation to be built without any knowledge about it in order to facilitate its future reuse, so it should be the aggregate responsibility to compose all the parts and to keep the track on them. We suggest stating this information only in the aggregate by using LOM item *hasPart*, thus avoiding the use of *isPartOf* in the parts. This approach forces systems to check the related resources to a given one that is composed by others, but preserves the individual reusability levels of simple educational resources. Besides all the mentioned issues, the fact that aggregation relationships entail *runtime commitments* that affect the objects that participate in the relationship has also to be considered. The most important of these commitments is *availability*, which means that the referenced resource has to be available whenever the current learning object is used or delivered. This can be done in two different forms: a) the referenced object being effectively available, or b) the learner providing evidence of a level of knowledge greater or equal than the stated in the learning outcomes published in the contract of A. Apart from availability, other commitments are *propagation* (some properties propagate from the aggregate to the parts), *acyclicness* (chains of aggregate links must not form cycles) or *reference validity*, a weak form of availability [13].

In the following example, a learning object *MyLO* that displays a Flash-animated example of the *Quicksort* algorithm, is in the list of candidates returned by the automated search engine, and it is consequently under consideration by a composer agent in order to integrate it in a Java course object:

```
rlo <http://... /MyLO>
 require
   mandatory   sys.browser >= V5_browser
   mandatory   sys.requirement = FlashPlugIn
   mandatory   ctx.cost = true ; recommended ctx.time = 0.5h
 ensure
   lrn.knows(qSort)[90]
```

The mentioned commitments affect the given object in the following way. *Propagation* requires the aggregate learning time to remain unknown until the total number of pieces is assembled, since it is calculated from them. *Availability* asks for MyLO to be publicly accessible whenever the aggregate is being delivered or used, but also for its usage fee to be paid. As MyLO is not composed by other learning objects (it is a leaf in the aggregation tree), *acyclicness* is guaranteed given that all the potential cycles would end here. So, if MyLO was finally chosen, the aggregate contract would be affected by the data in MyLO metadata record. The aggregate contract could be:

```
rlo <http://... /AgentGeneratedAggregate>
  require
    mandatory   sys.browser > V5_browser
    recommended ctx.time = 10h;  mandatory   ctx.cost = true
    mandatory   ctx.hasPart = "MyLO"     ...
```

```
ensure
  lrn.knows(qSort) [90]
```

As this example shows, MyLO design is not constrained by pre- or post- conditions in the aggregates it would be part of. It is the responsibility of the author of the aggregate to check the consistency of the parts with the aggregate contract, and to make use of some features to calculate the final values to be set in some assertions (as the *learning time* in our example). The new precondition *hasPart* appears after choosing MyLO and as a consequence of the relationship.

4 Example Architecture and Processing

In our model, contracts are the basis for searching and retrieving learning objects from a repository. We define *Learning Web Services* (LWSs) as Internet services that use a standard SOAP communication interface to expose learning objects that are described by metadata in the form of contracts. The LWS SOAP interface provides the clients with a) discovery of learning objects, based on their contracts, b) download of learning objects, using learning object packaging technologies [6], and c) learning object metadata retrieval. In what follows, a concrete design of LWS is described as a general-purpose architecture fulfilling the requirements of contract-based selection and composition. A declarative XL-like syntax [4] is used for the sake of obtaining a high-level architectural description.

When searching for learning objects, the query operation encodes a search request as a learning object contract archetype that will be in turn compared to the contracts of the RLOs stored in the repositories. Going back to our first example, let's suppose that an Italian-speaking learner is following a course of English as a foreign language; learning resources that are not locally available are required for the full course. As a lesson on the genitive case is needed, the LMS will need to locate a learning resource that fits to the current learner profile and system settings. A contract archetype will then be created from the course needs (which will form the postconditions in the contract) and limitations about the system and the learning context (preconditions in the contract), e.g.:

```
rlo <CONTRACT_ARCHETYPE>
  require
       sys.browser <= v5_browser;  ctx.time < 3h ; lrn.language = it
  ensure
       lrn.knows(genitive_case) [80]
```

This contract archetype asks for a RLO providing the learner with a knowledge on the genitive case over 80 percent confident. It limits the valid objects to those that can be displayed in a version 5 browser or lower, with a duration shorter than 3 hours, and intended to Italian-speaking audiences. A composer service can be used to provide the full selection and composition request through an interface like the following:

```
service <http://../Composer>
   history;
   let compositionRules rules;
   context let archetype arch;
   invariant archetype validates as contract;
   context let learningObject rlo;
   operation <http://../Composer>::compose
      precondition $input validates as contract;
      postcondition $output validates as learningObject;
```

```
      postcondition conforms($input, $output);
      body   ...
         <<divide in ci>>
         ci,rules -> <http://../ContractSearcher>::lookFor⇒rloi
         <<combine rloi>>
      endbody
   endoperation
endservice
```

The above XL-like syntax specifies a type called *archetype* for the `arch` context variable of the service, which is instantiated in a per-conversation basis. The `rlo` variable represents the ongoing composition. The type *compositionRules* is intended to reflect the global knowledge of the service regarding the constraints on composition required as those described in the previous examples. Such declarative specification can be easily expressed in XML, with syntaxes similar to those of RuleML[2]. The *history* clause entails the automatic recording of conversations; this is useful to cache repeating requests. The combination of RLOs entails the definition of a new aggregated learning object with metadata reflecting the contracts of the parts. In this model, not only RLOs can be retrieved, but also their contracts. Contract retrieval is particularly useful when choosing among RLOs offering similar outcomes, since the assessment criteria is the contract. Repositories of RLOs and contracts can be accessed through separate services with a minimal interface like the following:

```
service <http://../ContractRepository>
  operation <http://../ContractRepository>::query
     precondition $input validates as contract;
     postcondition conforms($input, $output);  ...
  endoperation
endservice
service <http://../RLORepository>
  operation <http://../RLORepository>::get
     precondition $input validates as LOMid;
     postcondition $output validates as learningObject;   ...
  endoperation
endservice
```

A software agent working on behalf of the end-user application (called Proxy agent) interacts with another software service called Composer. Composer takes an archetype contract from the expressed needs of the learner and will hand parts of it (c_i) over to searching services conforming to ContractSearcher. This intermediation allows for quality assessments and other criteria to be located on searchers, so that the repositories are simply persistent storages. ContractSearchers may also implement a subset of combination rules not included in Composer, allowing for extensibility of the approach to new sub-schemas. The requirements for searchers is that they should be able to provide an ordered collection of RLOs conforming to the given (sub-)contract, so that results are "explainable" according to the ordering criteria of searchers.

```
service <http://../ContractSearcher>
  let contractRepositoryCollection sources;
  let compositionRules rules;
  operation <http://../ContractSearcher>::lookFor
   precondition $input$ validates as contract, compositionRules;
   postcondition conforms($input[contract], $output);
    body
     $input$[contract]→ <http://../ContractRepository>::query → loi
```

[2] http://www.ruleml.org/

```
      <<sort lo_i>>   ...
    endbody
  endoperation
endservice
```

The communications between the Composer and the searchers is typically asynchronous, resembling Request-for-Quote business processes, while queries to repositories typically require a synchronous, immediate response. The decomposition just described carries out the essential composition and selection tasks. Additional supporting services can be added to deal with repository discovery and assessment. The complex conditions contracts impose on LO metadata require some form of logics-based support if maximum flexibility is desired. The *Web Services Modeling Ontology* framework[3] is an ideal candidate for such kind of effort. WSMO could be used to provide the description for *lookFor* implementation. An example fragment of an ontology definition in WSML could be:

```
concept learningObject
     nonFunctionalProperties
       dc#description hasValue "Any digital entity that may be used for learning,
       education or training"
     endNonFunctionalProperties
     aggregationLevel impliesType  (1 1) aggregationLevel
     languages impliesType humanLanguage
     isClassifiedInto impliesType classification
     hasRights impliesType rights
     hasTechnicalRequirements impliesType technicalRequirement
     locationURI impliesType _iri
     hasEducational impliesType  (1 1) educational
     identifier impliesType  (1 1) learningIdentifier
     title impliesType _string
     structure impliesType  (1 1) structure
     hasRelations impliesType relationship

concept educational
     nonFunctionalProperties
       dc#description hasValue "The Educational aspects of the Learning Object"
     endNonFunctionalProperties
     descriptionOfEducational impliesType _string
     interactivityType impliesType interactivityType
     learningResourceType impliesType learningResourceType
     hasInteractivityLevel impliesType interactivityLevel
     hasDifficulty impliesType difficulty
     contextEducational impliesType contextEducational
     intendedEndUserRole impliesType intendedEndUserRole

concept orCompositeTechnicalRequirement
     nonFunctionalProperties
       dc#description hasValue "Define the possibilities of a choice      for       a
       technical Requirement of a LO"
     endNonFunctionalProperties
     minimumVersionOfTR impliesType _string
     maximumVersionOfTR impliesType _string
     typeOfRequirement impliesType _string
     nameOfRequirement impliesType _string

concept technicalRequirement
     nonFunctionalProperties
       dc#description hasValue "A Technical Requirement that a LO has to comply"
     endNonFunctionalProperties
     installationRemarksOfTR impliesType  (0 1) _string
     formatOfTR impliesType  (0 1) _string
     sizeOfTR impliesType  (0 1) _string
     locationOfTR impliesType  (0 1) _iri
     durationOfTR impliesType  (0 1) _duration
     hasOrCompositeRequirements impliesType orCompositeTechnicalRequirement
```

[3] http://www.wsmo.org

```
concept learningIdentifier
      nonFunctionalProperties
        dc#description hasValue "An unique and unambigous identifier for a LO"
      endNonFunctionalProperties
      entryCatalog impliesType _string
      catalogIdentifier impliesType _string

concept resourceInRelation
      nonFunctionalProperties
        dc#description hasValue "Describes the Resource (LO) which is related with
        other LO"
      endNonFunctionalProperties
      descriptionOfResource impliesType _string
      identifiedResource impliesType learningIdentifier

concept kindOfRelation subConceptOf _string
      nonFunctionalProperties
        dc#description hasValue "Defines one kind of Relation between two LO"
      endNonFunctionalProperties

concept relationship
      nonFunctionalProperties
        dc#description hasValue "Describes one relationship between the LO owner and
        other LO"
      endNonFunctionalProperties
      kindOfRelationship impliesType kindOfRelation
      resourceInRelationship impliesType resourceInRelation

concept structure subConceptOf _string
      nonFunctionalProperties
        dc#description hasValue "Defines the structure in which is composed a LO"
      endNonFunctionalProperties

concept aggregationLevel subConceptOf _integer
      nonFunctionalProperties
        dc#description hasValue "Describes the granularity of the LO"
      endNonFunctionalProperties
```

Such shared formal definitions could be used to specify Web Services that provide a concrete kind of learning objects, as illustrated by the following WSML fragment that describes the capability to provide LO with concrete language, aggregation level and content classified according to certain taxonomy elements:

```
webservice _"http://www.uah.es/ontologies/ws.wsml"

nonFunctionalProperties
  dc#title hasValue "Algorithm for Internet Applications Learning Object Web Service"
  dc#description hasValue "Web service for access the content of a Learning Object on
  Algorithms and purchase it"
endNonFunctionalProperties

importedOntologies _"http:://www.wsmo.org/ontologies/purchase"

capability _#

 precondition
    axiom _#
    nonFunctionalProperties
    dc#description hasValue "The input to the Web Service has to be a user with an
intention to select a Learning Object"
    endNonFunctionalProperties
definedBy
   ?Buyer memberOf po#buyer.

  postcondition
    axiom _#
    nonFunctionalProperties
    dc#description hasValue "the output of the service is a Learning Object about
  Internet Algorithms."
```

```
    endNonFunctionalProperties
definedBy
?LO memberOf lom4WSMO#learningObject[
   isClassifiedInto hasValues {?Classifications},
   languages hasValues {lom4WSMO:englishUK},
   aggregationLevel hasValue 3
   identifier hasValue ?Identifier,
   title hasValue "Algorithms for Internet Applications (WS2001/02, lecture 14)"] and
?Identifier memberOf learningIdentifier[
   entryCatalog hasValue lom4WSMO#ARIADNE,
   catalogIdentifier hasValue "V3VIROR_v_3.1_nr_22"] and
?Classifications memberOf lom4WSMO#classification[
   purpose hasValue lom4WSMO#discipline,
   taxonPath hasValues {?Paths}] and
?Paths memberOf lom4WSMO#taxonPath[
   hasSourceTaxonPath hasValue lom4WSMO#ARIADNE,
   hasTaxon? hasValues {
      idTaxon hasValue "000000001",
      valueTaxon hasValue "Exact, Natural and Engineering Sciences",
      fatherOfTaxons hasValues {
        idTaxon hasValue "000000002",
        valueTaxon hasValue "Informatics & Information Processing",
        fatherOfTaxons hasValues {
           idTaxon hasValue "000000003",
           valueTaxon hasValue "General"}},
      idTaxon hasValue "000000004",
      valueTaxon hasValue "Internet Algorithms"}].

  effect
    axiom _#
    nonFunctionalProperties
    dc#description hasValue "there shall be a trade for the Learning Object of the
postcondition"
    endNonFunctionalProperties
  definedBy
    ?someTrade memberOf po#trade[
         po#items hasValues {?LO},
         Po#payment hasValue ?acceptedPayment]
    and ?acceptedPayment memberOf po#creditCard.
```

In order to match the user desire with the Web Service capability, that's the access to the learning object described in it, we need to define the desired learning facility required by the user by means of a WSMO Goal, as in the WSML fragment written below.

```
goal _"http://www.uah.es/ontologies/goals/goalLO.wsml"

nonFunctionalProperties
  dc#title hasValue "Searching for a Learning Object about Internet Algorithms"
  dc#description hasValue "Express the goal of buying a Learning Object for learn
Internet Algorithms"
endNonFunctionalProperties

importedOntologies {_"http://www.uah.es/ontologies/lom4WSMO"}

postcondition
axiom purchasingLearningObject
nonFunctionalProperties
  dc#description hasValue "This goal expresses the general desire of purchasing a
Learning Object"
endNonFunctionalProperties
definedBy
  exists ?LearningObject, ?Classification, ?Paths
  (
  ?LearningObject memberOf lom4WSMO#learningObject[
         lom4WSMO#isClasssifiedInto hasValue Classification?] and
  ?Classifications memberOf lom4WSMO#classification[
         taxonPath hasValues {?Paths}] and
  ?Paths memberOf lom4WSMO#taxonPath[
         hasSourceTaxonPath hasValue lom4WSMO:ARIADNE,
         hasTaxon? hasValues {valueTaxon hasValue "Internet Algorithms"}]
```

).

In consequence, both the decomposition of selection and composition in sub-activities and the expression of contracts in terms of formal ontology can be interpreted in terms of a contract-based approach in which goals and Web services logically match to serve the user with the desired set of Learning Objects required by his learning needs.

5 Conclusions

Learning object contracts can be used to drive the process of selection and composition of learning resources in a consistent way. Pre- and post-conditions can be used as search criteria, and aggregation relationships can be used to derive aggregate metadata. Such processes can be properly devised following a service-oriented approach, as illustrated by the high-level specification provided. Future work should detail the influence of each concrete metadata element in such processes, following the lines of recent work [16], and the process itself (or a set of alternative configurable processes) should be specified to guarantee a standardized, common and predictable behaviour. It should also deal with implementing intelligent agents to carry out the selection and composition processes, capable of changing their goals at runtime by introducing a more interactive approach.

Acknowledgements

The research reported in this chapter is funded by the European Commission 6th Framework (IST) Project LUISA (FP6-027149).

References

1. Anido, L. E., Fernández, M. J., Caeiro, M., Santos, J. M., Rodríguez, J. S., Llamas, M. Educational metadata and brokerage for learning resources. *Computers & Education*, 38(4), 351 – 374 (2002)
2. Blackmon, W.H., Rehak, D. Customized Learning: A Web Services Approach. In: *Proceedings of Ed-Media '03*, (2003)
3. Farance, F.: IEEE LOM Standard Not Yet Ready For 'Prime Time. IEEE LTTF Learning Technology Newsletter, 5(1) (2003)
4. Florescu, D., Grünhagen, A., Kossmann, D. XL: An XML Programming Language for Web Service Specification and Composition. In *Proceedings of the International World Wide Web Conference*, 7-11 (2002)
5. IEEE LTSC. Learning Object Metadata (LOM). IEEE 1484.12.1 (2002)
6. IMS Global Learning Consortium, Content Packaging Final Specification. (2001)
7. Kabel, S., Hoog, R., Wielinga, B.J.: Consistency in Indexing Learning Objects: an Empirical Investigation. In: *Proceedings of the Learning Objects 2003 Symposium*, 26-31 (2003)
8. Lave, J., Wenger, E. Situated Learning: Legitimate Peripheral Participation. Cambridge, UK: Cambridge University Press (1990)
9. Meyer B.: Object Oriented Software Construction, Prentice Hall, 331-410 (1997)

10. Pagés, C., Sicilia, M.A., García, E., Martínez, J.J., Gutiérrez, J.M.: On the Evaluation of Completeness of Learning Object Metadata in Open Repositories. In: *Proceedings of 2nd Int. Conf. on Multimedia, Information & Communication Technologies in Education,* 1760-1764 (2003)
11. Sánchez-Alonso, S., Sicilia, M.A.: Expressing Preconditions in Learning Object Contracts, In *Proceedings of 2nd International Conference on Multimedia, Information & Communication Technologies in Education*, 1656-1660 (2003)
12. Sánchez-Alonso, S., Sicilia, M.A. How Learning Object Relationships affect Learning Object Contracts: commitments and implications of aggregation. In: *Proceedings of Ed-Media'04* (2004)
13. Sánchez-Alonso, S., Sicilia, M.A. On the semantics of aggregation and generalization in learning object contracts. In *Proceedings of the 4th IEEE International Conference on Advanced Learning Technologies*, 425-429 (2004)
14. Sicilia, M.A., García, E. On the Concepts of Usability and Reusability of Learning Objects, International Review of Research in Open and Distance Learning, 4(2) (2003)
15. Sicilia, M.A., Sánchez-Alonso, S. On the concept of Learning Object "Design by Contract". In WSEAS Transactions on systems, 2 (3), 612-617 (2003)
16. Sicilia, M.A., Pagés, C., García, E., Sánchez-Alonso, S, Rius, A. Specifying Semantic Conformance Profiles in Reusable Learning Object Metadata. In: *Proceedings of the 5th International Conference on Information Technology Based Higher Education and Training* (2004)
17. Ternier, S., Duval, E. Web Services for the ARIADNE Knowledge Pool System. In: *Proceedings of the 3rd ARIADNE International Conference* (2003)