Competency-Based Pedagogical Wrapping

Yacine Atif
College of Information Technology
UAE University
United Arab Emirates
Yacine.Atif@uaeu.ac.ae

Abstract—**In this paper, we propose a novel learning-framework that is context dependent. We adopt a broad definition of learning context, encompassing learning domains and learner competencies. Context-based learning requires dealing with three major research thrusts: pedagogical categorization, learner modeling, and context matching techniques. The system architecture relies on a context matching engine and a set of pedagogical learning patterns to re-purpose learning objects according to contextual situations. A prototype is implemented on top of a common learning service registry, which supplies learning schema that map a given learning context and a learner profile, as well as instances of instructional learning services.**

Keywords-pedagogy; learning personalization; learning design; semantic Web; Web services; design patterns.

I. INTRODUCTION

Significant progress in reusability of learning resources have been made [1, 8, 4, 6, 2]. Learning however, is not just about contents but also about the process through which content is repurposed into personalized learning patterns [9, 5, 3], which map instructional contents into experiential learning activities. Yet, this external pedagogical know-how is not conveyed in a reusable manner to field experts and learning interface designers. Process-oriented learning (instead of content-based approaches) is facilitated in this paper through adaptive learning patterns.

Figure 1 shows a taxonomy of learning specification models, where learning operations are highlighted. At design-time, content is provisioned and related patterns are advocated. Learning patterns are encapsulated into learning schema to reflect a subject-related learning prescription. A learning schema models possible evolutions of learning styles and constraints throughout an instruction process. The annotation of learning resources to standard packages, such as IEEE LOM (Learning Object Metadata which specification is available at *http://ltsc.ieee.org*) and the publication of related active services used to retrieve those packages are performed in the implementation-time phase of learning specification. Finally, run-time specifications address interfacing issues to accommodate accessibility requirements, and provide means for context-acquisition to trigger further dynamic pattern reconfiguration.

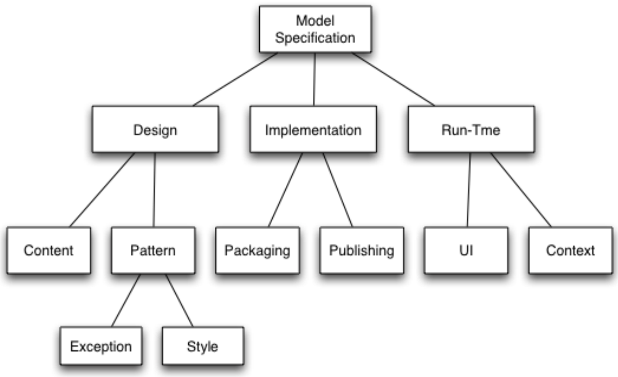


Figure 1. Learning Taxonomy

This paper proposes an approach to learning design which integrates learning patterns as standard learning services with tuning configuration parameters based on learner profiling criteria through standard LIP (Learner Information Package) specification. A learning process schema is introduced as a learning modeling capability to map contextual-dimensions via predefined pattern templates. The remaining sections of this paper are as follows: first, we provide some background and related work, which are relevant to the presented work in this paper.

In the remaining sections we first provide some background and state the problem addressed in this paper. Then we specify in Section 3 learning patterns as reusable educationalist-devised pedagofical templates, followed in Section 4 by learning designs to model learning patterns. We conduct an system prototype implementation in Section 5 as a proof of concept, and conclude the paper with a summary of results and suggested future directions.

II. BACKGROUND AND PROBLEM STATEMENT

Learning production is delivered in a standard layered-model shown in Figure 2. Typically, at the base level of the learning model is the domain model, which is an ontological structure to reason about the domain itself. It conceptualizes domain-related knowledge and defines semantic relationships, independently from pedagogical concerns. The subsequent layer of the learning model routes pedagogical goals and constraints via logical rules to operate over the domain model base layer. Goals refer to the sought-after competency levels, whereas learning-institutions, pedagogical requirements or personal learning-contexts may impose constraints.

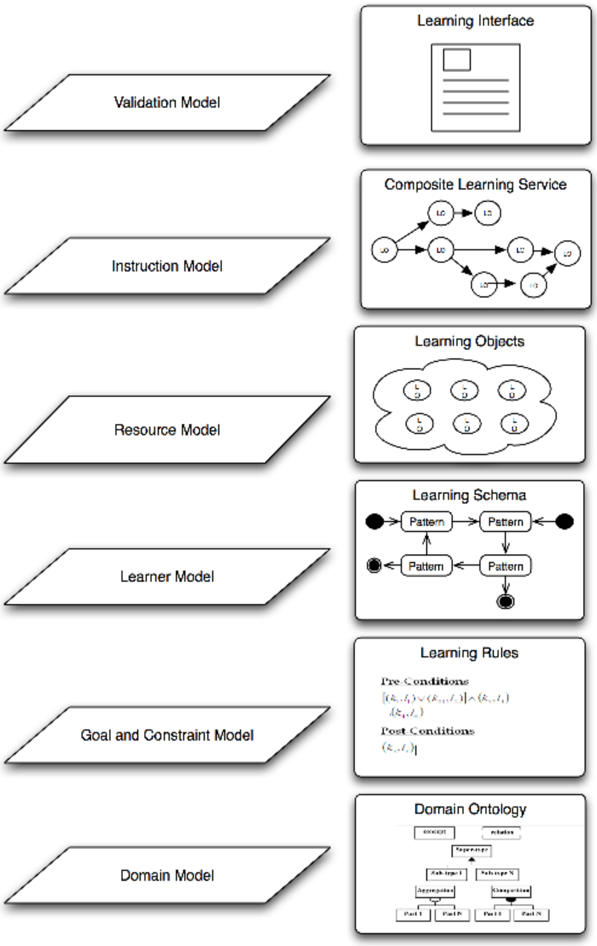


Figure 2. Learning Model

The third layer represents the learner model, which accumulates knowledge about learner profiles based on which, domain content can be wrapped into reconfigurable learning patterns. Pattern-specific learning tasks are prescribed at run-time to adapt to pedagogical recommendations and goals specification, based on pre-conditional rules consideration and post-conditional competency assessment. The learning resource layer shown in Figure 2 consists of learning services which access distributed repositories of Learning Objects or LOs. Learning services, implemented as standard Web services, subsume learning contents into standard packages defined by IEEE LOM specification. The course model choreographs a scenario of learning services as directed by a schema of learning patterns defined in this paper, through a service composition process.

The separation of functionalists into modular production units calls also for a separation of learning-provisioning roles, giving an opportunity to contribute at different levels of learning design. Learning content is tuned at each level to fit variety of profiles. Domain expert for example, may configure content to fit varying degrees of background. Pedagogues may instead mold learning content into activity patterns to adapt to a wider range of learning style. Instructional designers may package learning patterns into standard units and publish related services for learning distribution. Another possibility is for an HCI designer to advocate a presentation facade that is adjusted to specific learning services or customized to specific learning profiles.

Trends in providing personalized learning can be achieved by developing an open composition scheme of versatile learning-services provision within a multilevel enterprise integration framework such as the one shown in Figure 3. There is a lack of approaches today that enable open, enterprise-wide integration of learning, which empowers passive learning-structures such as LO to display a polymorphic behavior to fit learning patterns. In the learning architecture shown in Figure 3, lower layer modules are passive data structures that encompass raw learning-data at the bottom level, as well as structured learning resources and knowledge to package and reason about learning domain, at the upper level.

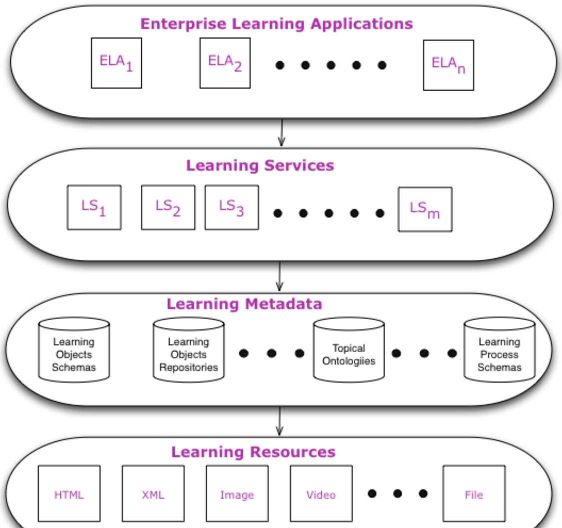


Figure 3. Enterprise Learning Architecture

III. LEARNING PATTERNS

As highlighted earlier, authors from different expertise horizons, like Content Experts, Instructional Designers, Pedagogues, etc, statically design instructional instances in each dimension of learning processes. In doing so, they retrieve an exiting instance of a given learning dimension and enumerate all possible instances across another dimension that fall onto their domain expertise. For example. A "Computer Science" pedagogue may consider an "Operating System" learning content and develop related activity patterns such as experiential learning cases through laboratories, pointers to discussion forums on the subject, problem-based practice lessons, etc. This authoring process may require an exhaustive enumeration of possible learning patterns, to match a wider range of typical learning profiles.

A. Learner Profile

Working out a learner's profile through some stereotypes can happen using the standard IMS Learner Information Package (LIP). This would permit to identify appropriate capability-enabled learning services. LIP is a specification of standard means for recording information about learners. It is designed to access information about learners, as well as their progress records. In doing so, LIP facilitates the transfer of learner-related information across different learning services or applications. As shown in Table 1, IMS LIP specification is structured in eleven groupings aimed at personalizing learning experiences in a general form. These groupings include Identification, Goal, QCL (Qualifications, Certifications or Licenses), Accessibility, Activity, Competency, Interest, Affiliation, Security Key, and Relationship.

TABLE I. LEARNER INFORMATION PACKAGE SPECIFICATION

Category	Explanation Explanation	Tag
Identification	Key bibliographic and demographic data about learner	<identification>
Goal	Career and other objectives and aspirations of an individual learner	<goal>
Qualifications	Qualifications, certifications and licenses granted by recognized authorities	<qcl>
Activity	Any learning-related activity in any state of completion	<activity>
Transcript	Summary of academic achievements	<transcript>
Interest	Hobbies and other recreational activities of a learner	<interest>
Competency	Skills, knowledge and abilities of a learner	<competency>
Affiliation	Learner's membership in different professional organizations	<affiliation>
Accessibility	Learning preferences as well as language capabilities, disabilities and eligibilities	<accessibility>
Security Key	Set of passwords and security codes assigned to a learner	<securitykey>
Relationship	Relationship between core data structures	<relationship>

Our approach exploits the inter-relationships between LIP elements to define learning patterns for an individual learner, within cognitive preferences, acquired competencies and expected learning goals, as shown in Figure 4. Competencies could be communicated in a suitable format to match learning patterns.

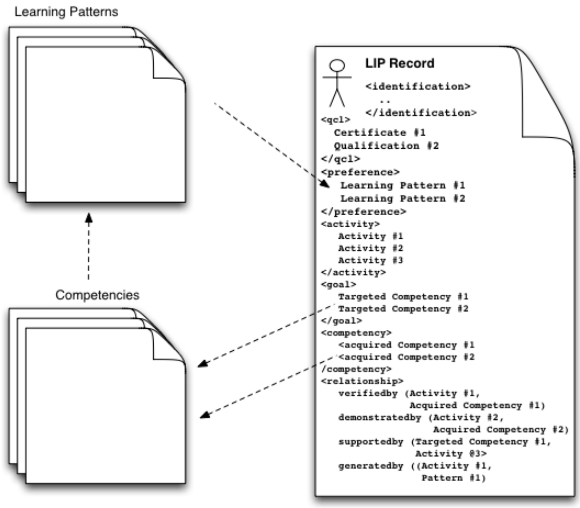


Figure 4. Learning Pattern

B. Learning Styles

Just as some people are left-handed, we learn better following different styles. A renowned educationalist David A. Kolb, has made an inventory of possible Learning Styles [7]. These learning modes are deemed to be responsive to contextual demands. According to Kolb, learners perceive and process information in a continuum from a concrete experience to testing implications as follows:

- Concrete Experience (CE): being involved in a new experience (feeling)
- Reflective Observations (RO): watching others or developing observations (seeing)
- Abstract Conceptualization (AC): creating theories to explain observations (thinking)
- Active Experimentation (AE): using theories to solve problems and make decisions (doing)

Depending upon the situation or the environment, learners may enter the above learning modes at any point. Below are some brief illustrative examples in applying Kolb's styles within different domain contexts:

- Abstract conceptualization - Listening to explanations and theoretical presentations.
- Concrete experience - Going step-by-step through an equation.
- Active experimentation — Practice on solving problems.
- Reflective observation - Recording thoughts about algebraic equations in a learning log.

Kolb's model offers both a way to understand individual learning styles, which he named the "Learning Styles Inventory" (LSI) [7], and also an explanation of a cycle of "experiential learning" that applies to all learners. Hence, our motivation to consider Kolb's model for defining learning patterns that can suit a variety of learning profiles. Kolb developed four learning styles: Diverger, Assimilator, Converger, and Accommodator. Learners generally prefer some of the four styles above the others or may move between styles in different learning domain contexts to reinforce retention. Kolb thought of these learning styles as a continuum that one moves through over time and hence, these are the main styles that instructors need to be aware of when creating instructional materials, particularly domain pedagogues. This instruction development approach results in four types of learning patterns:

1) Accommodators Pattern - (Concrete experience/Active experimenter pattern)

Learners who are exposed to this pattern are motivated by the question, "What would happen if I do this or that?" They look for significance in the learning experience and consider what they can do to move to the next stage. They like also to accumulate previous case scenarios elaborated by other learners. Example of instruction methods to create learning instances of Accommodators pattern include:

- Provide opportunities for independent discovery.
- Develop an active learning environment.
- Anticipate questions, such as "What if?" and "Why not?

2) Assimilators Pattern - (Abstract conceptualization/Reflective observer pattern)

Learners are eager to answer questions like, "What is there to know precisely?" They like accurate, organized delivery of information. They tend to pull a number of different observations and thoughts into an integrated whole. They also prefer to reason inductively and create models and theories before moving to design projects and conduct experiments. Instructional methods that suit Assimilators include:

- Lecture method (or video/audio presentation)—followed by a demonstration.
- Lab exploration, following a prepared tutorial for which answers should be provided.
- Independent study works and analytical exercises.

3) Convergers - (Abstract conceptualization/Active experimenter pattern)

For this type of learners, relevancy or the "how" of a situation are dominant learning-behaviors. Application and usefulness of information is increased by understanding detailed information about the system's operation. They tend to emphasize the practical application of ideas and solving problems. They like decision-making, processes and problem solving as well as practicability side of concepts. Finally, they prefer technical problems to interpersonal issues. Typical instruction methods that suit convergers include:

- Interactive instruction (i.e. not passive).
- Computer-assisted instruction.
- Exploring problem sets or workbooks.

4) Divergers - (Reflective observer/Concrete Experience pattern)

These learners are motivated to discover causality relevancy or "why" of a situation. They like to reason from concrete, specific information and to explore the information presented to them in a detailed, systematic and reasoned manner. They emphasize innovative and imaginative approaches in doing things. They view concrete situations from many perspectives and adapts by observation rather than by action. They like cooperative groups and brainstorming activities. Instructional methods that suit divergers include:

- Lecture focusing on specifics such as the strengths, weaknesses and uses.
- Hands-on examples. – Interaction sources with the instructor.
- Ready reference guides and organized summaries.

A learning profile may include several learning patterns. Content providers and pedagogues should use the above guide to develop learning packages. Each learning schema has one or more learning pattern. And each learning pattern belongs to exactly one learning schema. The line ended with a filled diamond between a learning schema and a learning pattern, denotes this composition relationship. This means, that a learning schema may encompass one or more learning patterns.

IV. Learning Design

We define a learning schema to encapsulate diverse learning pattern specifications, which are instantiated based on contextual information. Learning pattern models are behavioral approaches to learning. Learning schema is an overall structure that defines personalized learning processes. Learning schemas are reconfigurable to match the dynamic changes in learning patterns within a given learning experience. We represent a learning schema as a statechart, which is appropriate for modeling processes such as a learning process. Learning states are run-time reflections of learning patterns. As shown in Figure 5, a statechart is made up of states and transitions. States can be initial, end, basic, or compound. A basic state (representing in this paper a learning task) corresponds to an invocation of a learning service operation. Hence, patterns specify learning services and the service operations implement the pattern's learning tasks. Multiple operations may be invoked within a learning pattern, and several learning tasks may be invoked across learning patterns of a schema. Candidate learning services are selected based on their operations, which match a competency's functional requirements specified in a learning schema. A further selection among matching tasks is based on non-functional attributes such as QoS (Quality of Service) parameters or learning constraints to guide the matching process. Compound states in learning schemas provide a mechanism for nesting one or several statecharts inside a larger statecharts as shown in Figure 5. We use this abstraction to represent learning patterns as learning sub-processes within an overall learning process. This modeling technique facilitates dynamic transitions across learning states in a specific learning dimension, which may occur during a learning process. There are two types of compound states: OR and AND. An OR state contains single statecharts which represent alternative learning tasks, whereas AND states represent concurrent tasks separated by dashed line in Figure 5, which have to take place.

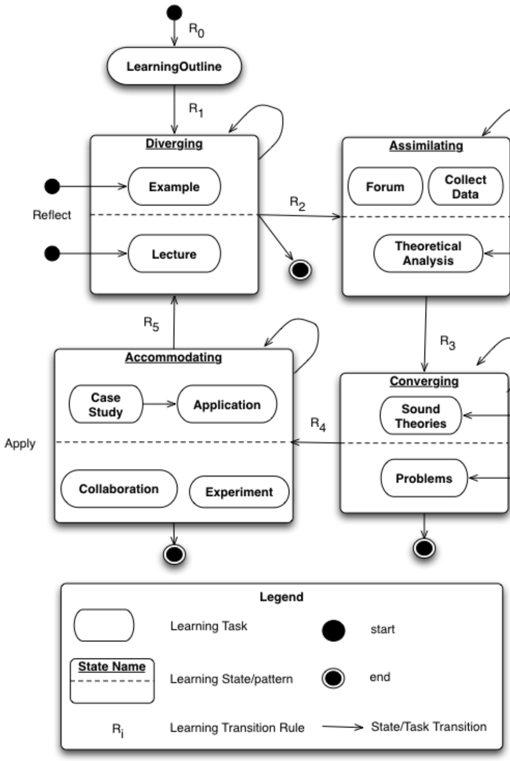


Figure 5. Learning Schema

Figure 5 presents an illustration of a simplified learning schema as a statechart diagram for a particular learning domain subject, which content have been designed. The elaboration of learning schemas may be devised by a pedagogue, in collaboration with the domain expert, or ideally by a domain-specific pedagogue, who could be seen as a senior domain expert with substantial accumulated experience in the subject instruction. In the example shown in Figure 5, learning may start from an overall outliner or from other learning unit instances. The closed circle shows the entry points leading to an initial state of a learning schema. The outliner task reveals the topical list of units modeled by this learning schema. The outliner task is followed by a transition to a Diverging pattern, where the learner may view hands-on examples concurrently with a focused lecture presentation. The learner may remain in the Diverging state for subsequent learning units (hence the loop at the compound box corner), or move to a new state. A transition to a new learning state is triggered by a condition and guided by a policy rule.

In this particular learning schema, the pedagogue advocates possible entry and exit points to fulfill the instruction outcomes. The bordered circle represents final states in the learning schema. For example, Diverging state may complete instruction requirements. However, assimilating state must be followed by converging state across a given learning unit to claim learning completeness in the subject matter, for which the schema in Figure 5 is advocated. Similarly, Accommodating state does not have an entry point (i.e. in this example, it cannot be the initial state). Hence, the learning schema is a learning guide rather than purely preferential style modeler.

Initially, learning schemas are designed and stored. A request for learning identifies a schema model. At run-time, matching learning services are matched to learning tasks in the schema. A coordinator agent sends the learning resources returned by the learning task to the context service along with the learning task policy. Learning reconfiguration is initiated by the learner context. Context data accompanies the reconfiguration event notification. The context-providing service verifies context data against learning-task post-conditions and updates the learner's (LIP) record of competencies accordingly. The next learning service invocation moves learner to a new state based on a schema of patterns description. If all patterns are exhausted or if learning services of the new pattern are unavailable, a new schema request is made.

V. PROTOTYPE IMPLEMENTATION

The prototype is implemented in Java on top of a common service registry, which supplies the learning schema as well as the learning service specifications. We used standard Web technologies such as XML, SOAP, WSDL and UDDI (Universal Description, Discovery, and Integration). Figure 6 shows of the prototype featuring the architecture of learning services as separate modules of objects. Learning Factory module generates a composite learning service to map a learning pattern specification and produces requested learning objects. Knowledge Factory module returns an XML specification of the learning schema, while the event-driven Context Factory module captures conditions for adaptations and triggers the reconfiguration process. Although simplistic, the prototype implementation is a proof of concept and a toolkit for usability evaluation. These three modules are further illustrated in Figure 6.

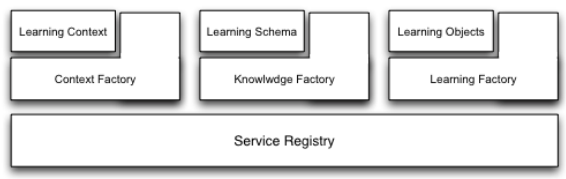
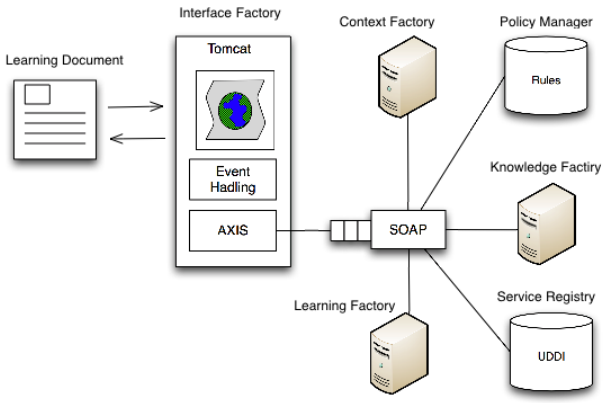


Figure 6. System Architecture

Services are deployed using Apache Axis. Apache Tomcat is used as a Web server where Apache Axis is deployed. Each service has a deployment descriptor that includes the unique identifies of the Java class to be invoked as well as the operations in the class that are available to clients. Learning Factory module uses a learning service registry, which is bound to learning objects repository. The service registry also includes a discovery and invocation facility implemented as SOAP calls (Figure 7). Learning services are simply used as a gateway to learning objects and may offer enhanced operations to deal with a learning object. When a learning service registers with a discovery engine, a UDDI SOAP request containing the service description in WSDL is sent to the UDDI registry. A learning service factory may deliver composite services from existing basic services in UDDI (for e.g. a case study followed by a collaborative project to match the Accommodating pattern description). The knowledge factory module accesses learning schemas stored in the UDDI as a tModels. tModel is a standard specification utility to represent a service type (a generic representation of a registered service) in the UDDI registry. Each learning service provider registered with UDDI maps all of its Web services according to a defined list of learning patterns within a schema (as specified in the example of Figure 5). An instructional designer can search the registry's learning schemas to create the enclosed pattern instances. The tModel organizes the service type's information and makes it accessible in the registry database. Each tModel consists of a name, an explanatory description, and a Universal Unique Identifier (UUID) . The tModel name identifies the service, such as, for example, "Computing Lecture" learning task.



Fi gure 7. Process Architecture

VI. CONCLUSION

The paper addresses learning personalization through adaptive patterns. We introduced learning schemas as means to design patterns of learning in a given instructional subject. Learning patterns encapsulate learning tasks, relevant to experiential learning processes, based on Kolb's foundations of learning theory. Learning schema specification use Statecharts and embedded patterns to reflect a learning style during instruction. Transitions across learning styles dictate learning reconfiguration through rules, which represent pedagogical requirements and personal constraints. Several directions could extend the presented work in this paper. First, we are currently building a portal application to enable learning producers at different level to participate in learning integration workflows. We could also export and share the accumulated learning services of advocated patterns to a public repository.

REFERENCES

[1] Iyad AlAgha and Liz Burd. Towards a constructivist approach to learning from hypertext. ACM Conference on Hypertext and Hypermedia, 2009. [

[2] Y. Atif, R. Benlamri, and J. Berri. Learning objects based framework for self-adaptive learning. Journal of Education and Information Technologies, Kluwer Academic Publishing, 8(4):345—368, 2003.

[3] Y. Atif, R. Benlamri, and J. Berri. Can e-learning be made real-time? Proceedings of the IEEE International Conference on Advanced Learning Technologie, ICALT'06, pages 314—318, 2006.

[4] Juan Blas, José Gutiérrez, Luis Marcos, and Roberto Barchino. Automatic e-learning contents composition by using gap analysis techniques. ACM SIGCSE Conference on Innovation and Technology in Computer Science Education, 2009.

[5] P Dolog, N Henze, W Nejdl, and M Sintek. Personalization in distributed e-learning environments. International World Wide Web Conference, 2004.

[6] C Hamel and D Ryan-Jones. Designing instruction with learning objects. International Journal of Educational Technology, 3(1), 2002.

[7] AY Kolb. The kolb learning style inventory˜version 3.1. Boston, MA: Hay Resource Direct, 2005.

[8] D Sampson and P Karampiperis. Towards next generation activity-based learning systems. International Journal on E-Learning, 2006.

[9] K Verbert, J Jovanovic, D Gasevic, and E Duval. Repurposing learning object components. Lecture Notes in Computer Science, 3762/2005(1), 2005.