Advanced Technology for Learning, Vol. 5, No. 1, 2008

CONTENT-DRIVEN DESIGN AND ARCHITECTURE OF E-LEARNING APPLICATIONS

C. Pahl*

Abstract

E-learning applications combine content with learning technology systems to support the creation of content and its delivery to the learner. In the future, we can expect the distinction between learning content and its supporting infrastructure to become blurred. Content objects will interact with infrastructure services as independent objects. Our solution to the development of e-learning applications – content-driven design and architecture – is based on content-centric ontological modelling and development of architectures. Knowledge and modelling will play an important role in the development of content and architectures. Our approach integrates content with interaction (in technical and educational terms) and services (the principle organization for a system architecture), based on techniques from different fields, including software engineering, learning design, and knowledge engineering.

Key Words

E-learning, learning technology systems, learning objects, software service architecture, ontologies, modelling

1. Introduction

Learning technology systems (LTS) are information technology-supported systems that are used to create, store, assemble, and deliver personalized learning content. LTS are concerned with the management of content, learners, and instruction. Content, often provided in the form of learning objects, and LTS together form e-learning applications. The development of these applications is a participative effort, involving software developers, content developers, instructors, and learners. It requires a combination of very different aspects: knowledge representation, software architecture, and learning and interaction process design. We take a content-centric view on the development of e-learning applications. While the separation of content and LTS has been beneficial in the past, in the future we can expect this distinction between learning content and its supporting infrastructure to become less obvious. Content is not only static material; interactive content will become standard and active content objects will interact with infrastructure services. The high costs involved in developing these interactive, multimedia-based learning content objects, however, requires a development approach for new architectural contexts that enables the reuse of content objects.

The architecture of e-learning applications is our focus. Software architecture concerns relate to higher abstraction levels of systems in terms of components and their dependencies. SCORM RTE (Sharable Content Object Reference Model, Runtime Environment) [1] standardizes interaction interfaces for basic e-learning systems architectures. It defines an interface for learning objects that interact with their LTS environment. Our objective here is to address architectures beyond SCORM RTE, aiming to bring the RTE context to a service- and knowledge-based level. This is a conceptual paper about development that addresses design and architectural principles, rather than platform and deployment aspects.

The systematic development of content and LTS has already been widely addressed [2–6]. The use of software engineering practices has strongly influenced these approaches. Another influence on the development of e-learning applications are standards in educational technologies such as the SCORM suite of standards. Our solution to the development of these systems – content-driven design and stepwise architecture development – is based on content-centric ontological modelling and architecture transformation. The aim of our approach is to reflect current trends in e-learning application architectures in an adequate development technique that is cost-effective through transformations of ontological models into architectures and that allows us to provide learning objects as interoperable and reusable services. Knowledge and modelling will play an important role in the development of content and architectures. Our approach integrates content with interaction in both technical and educational terms and services as the organizational principle of e-learning system architectures. We integrate methods and techniques from different fields into our approach: software engineering (modelling and software architecture), learning (instruction and interaction), and knowledge engineering (ontologies). We have developed these methods and

* Dublin City University, School of Computing, Dublin 9, Ireland; e-mail: Claus.Pahl@dcu.ie
(paper no. 208-1056)

techniques based on our own experience with intelligent tutoring systems (ITS) and ontology-based content management architectures. An ITS-style e-learning system, used to support a database course, is our case study.

2. Problem Definition

Content is an important element of e-learning applications. Knowledge and the interaction of learners with content – essential aspects of learning content design – are factors that drive the current research in this field. While in traditional systems, content is separated from the surrounding infrastructure, we take another perspective, seeing content objects as services equal to infrastructure services. This paradigm change follows the view taken in service-oriented software architectures [7] and middleware systems [8] where user-defined objects are technically not different from system services and facilities.

ITS are examples of e-learning applications where the distinction between content and infrastructure becomes blurred, as their main objective is to provide tutoring support for subject-specific content. With this view on ITS, Virvou and Tsiriga [5] apply classical object-oriented software development techniques to develop and maintain these systems applying an object notion throughout. However, the development of e-learning technology and content is currently driven by factors that have not been addressed in a coherent framework [2–6]:

- *Cost reduction and reuse.* Content is expensive to produce – reusable content units and services are therefore sought. Current standards such as SCORM, however, have focussed on metadata and packaging, but not on the development itself. Boyle [2] proposes design principles derived from pedagogy and software engineering, such as cohesion and de-coupling, to enable reuse by composing individual objects. Pahl [4] focuses on management and maintenance activities to reduce costs, using an analysis model to identify maintenance issues and to devise strategies to address these.
- *Interoperability and service-orientation.* Service-orientation, in particular using the Web as the platform, enables higher degrees of interoperability. A service provides reusable functionality at a certain location. Devedžić [3] describes service-based LTSs using ontologies (sharable knowledge representation frameworks) to model service capabilities.
- *Knowledge modelling and management.* Integrated knowledge management is an aim pursued in numerous organizations that is also relevant for education providers and learners. Pantano Rokou *et al.* [6] have based their approach to modelling the pedagogical knowledge for educational systems on an extension of the unified modelling language UML, which is widely used in software modelling [9]. Their educational model includes the student profiles, the educational goals, and the pedagogical strategies.

These current developments are driven by different factors affecting different aspects of e-learning applications. Reusing and organizing content to improve content management is a major objective. Separating knowledge from content can support the discovery of reusable content resources. Interoperability is required to integrate existing resources into a new environment. Architectures based on reusable and composable objects are currently the best contender for the interoperability problem. For the development of e-learning applications this means focusing on three modelling aspects: content, architectures, and interaction processes. As our focus is not the development of learner or content management infrastructure, but the development of applications, our starting point is content. Content objects shall be represented in form of services, embedded into interaction processes.

- Content is created by converting knowledge from various sources into learning objects. Modelling knowledge for interactive content is a current research problem. Content is based on two knowledge domains: subject and instruction. Process-centric ontologies [10], i.e., knowledge representations that capture the subject with its concepts and activities and instruction with its interaction and learning behaviour, provide a solution.
- Service-oriented architectures (SOAs) enable the desired degree of interoperability and reuse [7]. Static content can be combined with delivery services; interactive learning objects form independent services. Particular attention has to be paid here to the learner interaction processes involving interactive content services.

Although deployment and platform considerations are not central in our approach, we briefly look at the SCORM RTE standard [1], which defines an interaction infrastructure for independently deployable learning objects. RTE defines an interface that describes how a learning object can interact with an LTS. The interface addresses object lifecycle control, i.e., the interface allows a learning object to be launched at the beginning of its lifecycle and to be destroyed at the end, and activity logging, i.e., the learning object can pass variables and their values to the LTS in order for those variable/value pairs to be stored. Although far away from the architectural requirements we expect, RTE shows the trend towards more interactive learning objects as computational entities. Our aim here is an architectural framework beyond SCORM limitations.

3. Design and Development

Model-driven development [9] is a software engineering method that focuses on the development of systems starting with domain-specific models. Domain modelling, i.e., capturing knowledge about domains in models, is a central design activity. Software architecture is about the organization of software systems in terms of interacting components or services based on domain models. We propose domain modelling in the form of sharable ontologies and SOA as our design method. Subject domain and instruction knowledge forms the basis of content. In the architecture, we distinguish infrastructure and content-oriented services. Processes describe interaction between services, but also learning behaviour as a learner's interaction with content.

3.1 Knowledge and Content Modelling

Interaction with content is an important element of instructional design for e-learning. It determines the system services involved and the interaction processes that services are involved in. Learning, however, is not only a technical process; it is embedded into an educational and subject-specific context. Ontologies shall provide the modelling format for this context [10]. Ontology-based models can be used to specify requirements or services. Ontologies combine descriptional aspects of modelling languages like UML with analysis and reasoning capabilities. Ontological reasoning can analyse model consistency and support the discovery of reusable matching content in repositories of content descriptions [11, 12].

- The *subject domain* is captured in terms of its central concepts, i.e., objects and processes. The subject domain for our database case study is characterized by the following concepts: domain objects (such as *table*, *record*, *value*, *attribute*), domain processes (such as *create*, *delete*, *insert*, *query* as basic activities and process combinators such as *sequence* ";", *choice* "+", *iteration* "!", *concurrency* "||"), and domain rules (such as process ordering constraints on activities like *create;* !(*insert*+*query*); *delete*).
- The *instructional knowledge* can be structured into instruction objects (such as *definition*, *example*, *exercise*, *assessment*), instruction activities (such as *knowledge acquisition*, *observation*, *training*, *problem solving*, *knowledge creation*, *reproduction* which can be combined using *sequence*, *choice*, *iteration*, and *concurrency*), and instruction rules (e.g., to enforce the consistency of several units, e.g., *concept*(*definition*) = *concept*(*example*) = *concept*(*exercise*) for a given content unit, or sequencing constraints on objects like *example*; *exercise*; *assessment*).

Both subject and instruction models, see excerpts in Fig. 1, are activity-oriented models (*insert* and *query* are activities facilitated by services that process *record* and *table* objects in the subject model; *sequence* is an activity combinator for the *lecture* and *lab* activities in the instruction model). Both subject and instruction involve activities that relate to the processing of objects. Consequently, we focus on activity- and service-based processes in the ontological models. We are interested here in the input-to-output processing functionality of services. The process-centric ontology language we use is based on graphs, see Fig. 1, where e.g., the process *insert* requires concept objects *table* and *record* as inputs and produces a *table* object.

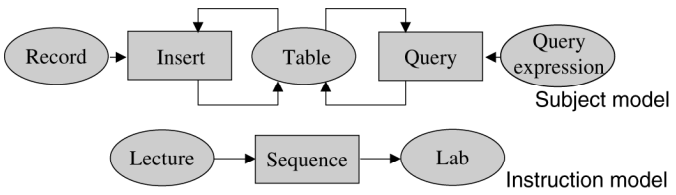

Figure 1. Process-oriented content ontology based on separate subject and instruction models.

3.2 Architectures, Services, and Processes

Some architectural frameworks based on SOA principles [7] are relevant for service-based e-learning systems. Looking at architectural concerns addresses key development problems outlined earlier. The learning technology system architecture (LTSA) is a reference architecture that can act as an infrastructure definition for learning objects [13]. If learning objects are dynamic interactive services, as we aim at here, the LTSA needs to be complemented. The CORBA middleware architecture standard [8] sees all computational elements as objects or components, which includes interactive content services and infrastructure services in our understanding. In the Web service framework (WSF) [7], all computational elements are services; services can be looked up in repositories and used in a distributed environment. These principles apply to both content (as learning content services) and infrastructure (as learning infrastructure services). Semantic Web services, an ontology-based enhancement of the WSF, combine modelling and architecture aspects. Based on these frameworks, we propose a design approach with services as basic architectural building blocks for both content and infrastructure.

Our generic architecture for e-learning applications is presented in Fig. 2. We distinguish infrastructure and content services. *Content services* can be divided into two subcategories:

1. *Content delivery services*: The media types involved usually indicate the service type for delivery. Content delivery services are media-specific services that range from the delivery of static content to interactive, multimedia-based content applications. To achieve interoperability, all services have to implement a common interface that allows their management by infrastructure services.

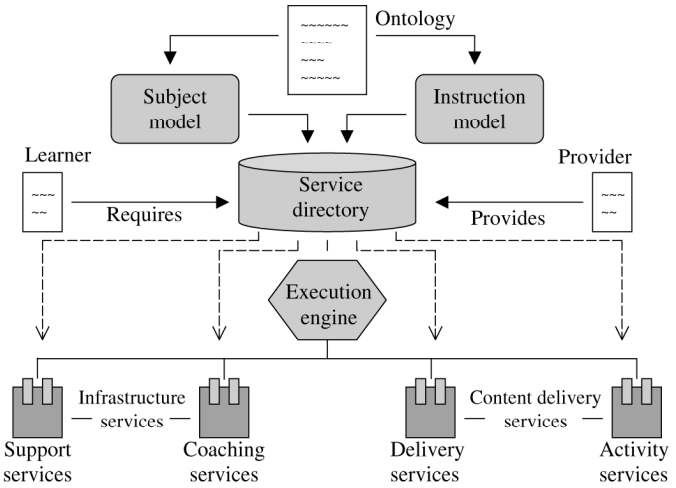

Figure 2. Generic service-based e-learning development architecture.

Table 1
Sample Services for a Database Course with Subject, Instruction, and Service Type

Service	Subject Aspect	Instruction Aspect	Service Type
SQL lab: text (table, query) to text (table) processing	SQL select with activity (query) and object (table)	*Training*: sequence, iteration, and choice of units	*Content activity service*: interactive, SQL input processing
SQL lab feedback: text (table and query) to text (feedback) processing	SQL select with activity (query) and object (table)	*Feedback*: error classification and description, hints, and solutions	*Coaching service*: individual feedback based on input
SQL tutorial: simulation (text and video)	SQL select with activity simulation	*Observation*: sequence and iteration of units	*Content delivery service*: synchronized media streaming

2. *Content activity services*: The more interactive, the more subject-specific these services become due to the increased level of content-specific input processing and delivery. Activity services support domain-specific learning activities.

Infrastructure services are also divided into two subcategories:

1. *Coaching services*: These are part of the scaffolding. A coach aims to match learning goals and suitable content. A coach deals with content composition and sequencing. It acts as a delivery supervisor. The more interactive, the more instruction involved this becomes due to more complex learning processes in interactive environments.
2. *Support services*: LTSs usually support a range of standard services such as assessment and evaluation services (e.g., student tracking), management services (e.g., account set-up, delivery service registration, content upload, and integration), and communication services (e.g., e-mail and discussion forums).

We focus on content and its coached delivery – neglecting basic infrastructure support services as provided by standard LTSs. In Table 1, we have classified sample services from the databases context. The combination of subject and instruction knowledge determines the service type and the delivery mechanism. Smaller examples are content services; more complex services include coaching elements as well, which need access to additional resources such as solutions and learner models. Classifying and describing services in this form is the next step in the design of an e-learning application after the modelling of domain and instruction knowledge in ontological form.

3.3 Stepwise Development and Transformation

The ontological subject and instruction models, see Fig. 1, have to be mapped to the services we just identified in a service-oriented e-learning system architecture. We ignore here the coaching, management, and communications services as target services and focus on content services. Mapping is a stepwise transformation process starting with the subject and instruction models and ending with an assembly of described and implemented services.

- *Step 1: Service identification and classification.* We distinguish two types of content services that can be identified in the two models:
  1. *Content delivery services*: Subject and instruction objects are combined; content is delivered through standard services (e.g., streamed audio for knowledge acquisition). The delivery is media-specific and, therefore, usually content-independent. These services are connected only through the instruction processes.
  2. *Content activity services*: Subject activities, embedded into an educational context by an instruction object, are represented in the form of a service that provides this activity (real or simulated, e.g., database queries, for skills training or assessment). In contrast to the content delivery services, these services facilitate the actual activities.

Table 1 lists classified services and their relationship to subject and instruction models.

- *Step 2: Service description and implementation.* Services need to be described to enable their implementation and reuse. We distinguish domain objects and activities.
  1. Content based on static objects of the subject domain can often be delivered using existing services (these might have to be discovered in repositories). The service requirements are determined by the media type. The functionality of these services is based on a content identifier as input and the delivered content as output.
  2. Content that support subject activities can be developed as services from scratch or reused from a repository. These services process objects from the subject domain, i.e., transform subject objects as input into output objects based on the activity.

The model-to-architecture transformation is a mapping from process concepts to some service that processes objects of some kind.

- *Step 3: Process assembly.* Processes are composed services. Processes are often based on subject-specific rules for service composition (e.g., database operations lifecycle) or on rules for instructional composition (e.g., an instructional sequence with definition, example, and exercise). Both subject and instruction rules form constraints that have to be obeyed when subject and instruction processes are merged. Consider *lecture*; *lab* as the instructional process and *create*; *insert*; *select* as the subject-specific process.

$$\begin{array}{ccccc} create\_lecture & \rightarrow & insert\_lecture & \rightarrow & select\_lecture \\ \downarrow & & \downarrow & & \downarrow \\ create\_lab & \rightarrow & insert\_lab & \rightarrow & select\_lab \end{array}$$

Suitable merged processes would be line-by-line, left-to-right, and column-by-column, top–down – both satisfy the ordering constraints expressed in the process specifications.

In the database case study, the *lab* service is characterized by two input and two output elements. The first input element is a *query* (a formal language expression, representing an activity), the second is a database *table* (another formal expression, representing an object). The first output element is the resulting *table*; the second is *feedback* (a semiformal expression, representing instructional advice in the form of error classification and guidance).

4. Discussion of the Design and Architecture Approach

In traditionally organized e-learning applications, content is separated from infrastructure for its creation, management, and delivery. Like us, a number of authors, e.g. [12, 14], argue that in the wake of service-orientation, learning objects, and interactive learning content this distinction will become blurred. With widening acceptance of service-orientation, the infrastructure support, previously provided through learning content and learner management systems, can be expected to be available in the form of independent, composable services [14]. Web services are meant to provide an interoperable implementation platform [12] – although our investigation is not about implementation, but rather about the challenges facing this domain and how to address them. A systematic approach to the development of service-based e-learning applications is needed to cater for the new platform technology.

Our approach to support content and infrastructure developers is based on techniques proven to enhance the quality of the architectures of these applications. We summarize the experience that we, but also others have made in terms of important design and architecture aspects – see for instance [15] for a comprehensive account.

- *Development.* Model-driven development as the overall development methodology is proven to reduce development time and costs and to improve maintainability [9] – the latter is an often-neglected aspect, which we have addressed in more detail in [4].
- *Reusability.* Ontology-based modelling enables reuse and consequently cost reduction and quality improvement, which we have documented for a knowledge-based learning content management system for learning objects [16].
- *Modularity.* Architectural design is beneficial through modularity, achieving flexibility and maintainability, as e.g., Brusilovsky [17] and Yang *et al.* [12] have also shown.
- *Personalization.* Explicit knowledge, used by us in the form of ontologies, is a prerequisite for adaptivity [18], enabling the personalization of content and delivery.

A comprehensive implementation of our approach is currently limited by the degree of analysis and transformation automation, which are central for increased cost-effectiveness. Nonetheless, the methodology has provided us with a structured approach to develop a quality system architecture. The methodology is effective in providing interoperability based on the service platform and maintainability based on reusability and modularity methods, as the re-engineering of the case study system has demonstrated [4, 16].

5. Conclusions

With the proliferation LTSs and the increasing costs for content development, the need to address the design and implementation of these systems and their content as e-learning applications from a software engineering perspective, but also within the application context, is imminent. Two development aspects characterize this context:

- Standards have started to impact the development of learning content and systems. The SCORM standards describe metadata, packaging, sequencing, and runtime aspects. With the Web as the predominant platform and reuse becoming a central objective, Web services are likely to be the architectural platform of future e-learning applications. The LTSA is a first move to describe these systems as a collection of interacting services.
- Ontologies have been used in educational technology to capture and represent knowledge in relation to the subject but also instruction domain. We have proposed an approach similar to the currently discussed model-driven development. Ontological models form the starting point that will then be mapped to services and service interaction processes.

We have captured these developments in a content-driven design and architecture approach for e-learning applications. Models and architectures are the central cornerstones of our development approach. The technology is currently moving towards a service-oriented platform. With infrastructure services becoming available and standardized, we have argued that content and its provision through services should be at the centre of our attention. Content is the application-specific part of service-based

e-learning system architecture. Interaction processes facilitating the interaction between learners and content is important in these architectures. With individual and composable infrastructure services becoming readily available, these can then be assembled around content services. Collaboration – an aspect that we have neglected here, but which needs consideration due to its focus – can also be addressed based on services layered on top of content and infrastructure services.

References

[1] Advanced Distributed Learning ADL, SCORM Sharable Content Object Reference Model, 2004, http://www.adlnet.org/index.cfm?fuseaction=scormabt (visited 16/05/2006).

[2] T. Boyle, Design principles for authoring dynamic, reusable learning objects, *Australian Journal of Educational Technology*, *19*(1), 2003, 46–58.

[3] V. Devedžić, *Semantic web and education* (Berlin: Springer-Verlag, 2006).

[4] C. Pahl, Managing evolution and change in web-based teaching and learning environments, *Computers and Education*, *40*(1), 2003, 99–114.

[5] M. Virvou & V. Tsiriga, An object-oriented software life cycle of an intelligent tutoring system, *Journal of Computer Assisted Learning*, *17*, 2001, 200–205.

[6] F. Pantano Rokou, E. Rokou, & Y. Rokos, Modeling web-based educational systems: Process Design Teaching Model, *Educational Technology and Society*, *7*(1), 2004, 42–50.

[7] G. Alonso, F. Casati, H. Kuno, & V. Machiraju, *Web services – concepts, architectures and applications* (Berlin: Springer-Verlag, 2004).

[8] C. Britton, *IT architectures and middleware: Strategies for building large, integrated systems* (Boston, USA: Addison Wesley, 2004).

[9] S. Mellor, A. Clark, & T. Futagami, Model-driven development, *IEEE Software*, *20*(5), 2003, 14–18.

[10] M.C. Daconta, L.J. Obrst, & K.T. Smith, *The semantic web* (Indianapolis, USA: Wiley & Sons, 2003).

[11] D.G. Sampson, M.D. Lytras, G. Wagner, & P. Diaz (Eds.), Ontologies and the semantic web for E-learning, *Journal of Educational Technology and Society*, *7*(4), 2004, 26–28.

[12] S.J.H. Yang, B.C.W. Lan, I.Y.L. Chen, B.J.D. Wu, & A.C.N. Chang, Context aware service oriented architecture for web-based learning, *Advanced Technologies for Learning*, *2*(4), 2005, 216–222.

[13] IEEE Learning Technology Standards Committee LTSC, *IEEE P1484.1/D8, Draft Standard – Learning Technology Systems Architecture (LTSA)*, IEEE, 2001.

[14] P. Karampiperis & D. Sampson, Designing learning services: From content-based to activity-based learning systems, *Proc. 14th International World Wide Web Conf. WWW2005*, Chiba, Japan, 2005, 1110–1111.

[15] C. Bouras & T. Tsiatsos, Educational Virtual Environments: Design Rationale and Architecture, *Multimedia Tools and Applications Journal*, *29*(2), 2005, 153–173.

[16] C. Pahl, E. Holohan, M. Melia, & D. McMullen, Ontology-based learning objects in learning content management systems, in A. Koohang & K. Harman (Eds.), *Principles and practices of the effective use of learning objects* (Santa Rosa, USA: Informing Science Press, 2006).

[17] P. Brusilovsky, Knowledge Tree: A distributed architecture for adaptive e-learning, *Proc. 13th International World Wide Web Conf. WWW2004*, New York, USA, 2004, 104–113.

[18] P. de Bra, Adaptive educational hypermedia on the web, *Communications of the ACM*, *45*(5), 2002, 60–61.

Biography

Claus Pahl is a Senior Lecturer and the leader of the Web and Software Engineering research group at Dublin City University, which focuses on Web technologies, software architecture, and e-learning applications in particular. Claus has published more than 150 papers including a wide range of journal articles, book chapters, and conference contributions on e-learning and software architecture. He is a reviewer for journals and member of programme committees for various conferences. One of his central research activities focuses on semantic modelling and service architectures. He has extensive experience in educational technologies, both as an instructor using technology-supported teaching and learning at undergraduate and postgraduate level and as a researcher in Web-based learning technology. The IDLE environment, developed by him and his students, is in use in undergraduate teaching since 1999.