TECHNISCHE UNIVERSITÄT DARMSTADT

# Modularization and Multi-Granularity Reuse of Learning Resources

Vom Fachbereich
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des Grades eines
Doktor-Ingenieurs (Dr.-Ing.)

**genehmigte Dissertationsschrift**
von

**Dipl.-Inform. Marek Meyer**

geboren am 3. September 1979 in Bad Soden am Taunus

Vorsitz: Prof. Dr.-Ing. Volker Hinrichsen
Erstreferent: Prof. Dr.-Ing. Ralf Steinmetz
Korreferent: Prof. Dr.-Ing. Abdulmotaleb El Saddik

Tag der Einreichung: 10.07.2008
Tag der Disputation: 26.09.2008

D17
Darmstadt 2008

## Kurzfassung

Seitdem Personalcomputer in Büros und Haushalten alltäglich geworden sind, haben Anwender auch damit begonnen diese Rechner für Bildungszwecke einzusetzen. Der Begriff E-Learning wird als Oberbegriff für alle Arten von Lernszenarien verwendet, die computergestütztes Lernen oder Lehren einbeziehen. Für den Lernenden ergeben sich oftmals neue Freiheiten: er kann wählen wann, wo und wie er lernt. Lernressourcen sind digitale Materialien (z.B. Dokumente, Bilder, Videos, Simulationen, Tests, etc.), die in Lernszenarien genutzt werden. Wenn Lernen in einem formalen Kontext stattfindet, muss ein Lehrender üblicherweise diese Lernressourcen den Lernenden zur Verfügung stellen.

Die Erstellung von Lernressourcen durch die Lehrenden selbst oder durch Dritte ist Forschungsgegenstand vieler Untersuchungen. Es ist mittlerweile allgemein anerkannt, dass die Erstellung qualitativ hochwertiger Inhalte arbeitsintensiv ist und daher hohe Kosten verursacht. Daher ist die Wiederverwendung bereits existierender Lernressourcen erwünscht um Kosten zu reduzieren bzw. zu vermeiden. Neben der unmittelbaren, erneuten Nutzung von Lernressourcen zum Lernen oder Lehren in neuen Kontexten können existierende Lernressourcen auch als Vorprodukt für die Erstellung neuer Lernressourcen in Betracht gezogen werden. Ähnlich der Wiederverwendung von Softwarebibliotheken in der Softwareentwicklung erwartet man von der Wiederverwendung von Lernressourcen als Vorprodukt eine Senkung der Produktionskosten neuer Lernressourcen. Da E-Learning heutzutage in universitärer Lehre, Fort- und Weiterbildung sowie im Produkttraining sehr beliebt ist, fällt der Erstellung von Lernressourcen eine große Bedeutung zu.

Die vorliegende Dissertation bezieht sich insbesondere auf ein Wiederverwendungsszenario, in dem existierende Lernressourcen als Vorprodukte für die Erstellung neuer Lernressourcen zum Zwecke des Web-basierten Lernens dienen sollen. Autoren sind an der Verwendung von Lernressourcen anderer Autoren interessiert. Es wird angenommen, dass diese Autoren unterschiedlichen Organisationen angehören. Des Weiteren können sich die Autoren nicht auf ein gemeinsames Autorenwerkzeug einigen, weil jeder Autor verpflichtet ist die von seinem Arbeitgeber vorgegebenen Werkzeuge zu verwenden. Der Austausch von Lernressourcen zwischen diesen Autoren wird daher zur Herausforderung. Es gibt verschiedene Inhaltsmodelle, die den hierarchischen Aufbau von Lernressourcen beschreiben. Autorenparadigmen, wie beispielsweise ein auf Aggregation basierender Autorenprozess, ermöglichen grundsätzlich die Erstellung neuer Lernressourcen als Kombination kleinerer Lernressourcen. Dafür ist es jedoch erforderlich, dass die zu kombinierenden Lernressourcen als eigenständige Ressourcen gespeichert werden. Dieser Ansatz funktioniert gut, solange eine Organisation systematisch feingranulare, modulare Lernressourcen mittels einer dafür geeigneten Autorenumgebung erstellt. Viele Autorenwerkzeuge nutzen arbiträre Inhaltsformate, die inkompatibel zu denen anderer Autorenwerkzeuge und weiterer Systeme (beispielsweise Lernmanagementsysteme) sind. Aus diesem Grund werden diese Formate nicht für den Austausch von Lernressourcen verwendet. Stattdessen werden Lernressourcen in das Shareable Content Object Reference Model (SCORM) Format konvertiert. SCORM ist der de facto Standard für Web-basierte Lernressourcen, der von nahezu allen Autorenwerkzeugen und Lernmanagementsystemen unterstützt wird. Ein

Nachteil dieses Vorgehens ist, dass die ursprünglich modularen Komponenten einer Lernressource nach der Konvertierung nicht mehr als eigenständige Lernressourcen verfügbar sind.

Diese Dissertation ermöglicht die modulare Wiederverwendung von Lernressourcen, die aufgrund von Konvertierungen nicht länger als individuelle Lernressourcen existieren. Zwei Definitionen bilden die Grundlage dieser Arbeit: Modularität in Bezug auf Lernressourcen und deren Formatspezifikationen, sowie multi-granulare Wiederverwendbarkeit von Lernressourcen. Die zugrundeliegende Definition von Modularität erfordert, dass Komponenten einer Lernressource gekapselt, nach außen sichtbar gemacht und als separate Module wiederverwendet werden können. Multi-granulare Wiederverwendbarket erweitert das Modularitätskriterium durch die Forderung, dass jeder relevante Bestandteil einer Lernressource in dieser Weise wiederverwendet werden kann.

Das Ziel der multi-granularen Wiederverwendbarkeit wird durch die Aufstellung von sechs Kriterien operationalisiert. Die einzelnen Beiträge dieser Dissertation müssen sich an diesen Kriterien messen lassen. Diese Kriterien sind einerseits drei technische Anforderungen der multi-granularen Wiederverwendbarkeit – *Verfügbarkeit*, *Auffindbarkeit* und *Interoperabilität* - und andererseits drei modulare Operationen, die für die Wiederverwendung benötigt werden: *Modularisierung*, *Aggregation* und *Anpassung*. Die vorliegende Dissertation weist fünf Beiträge auf, die jeweils eines oder mehrere der genannten Kriterien unterstützen. Im Zusammenspiel tragen diese fünf Beiträge zum Ziel der multi-granularen Wiederverwendbarkeit von Lernressourcen bei.

Der erste Beitrag besteht in einer Erweiterung der SCORM-Spezifikation, welche die modulare Wiederverwendung von Komponenten eines SCORM-Pakets, sowie die Modularisierung und Aggregation von Lernressourcen ermöglicht. Des Weiteren wurden mehrere Ansätze zur Modularisierung von Lernressourcen untersucht. Als Ergebnis wurde ein generisches Prozessmodell für die Modularisierung von Lernressourcen erstellt. Dieses Prozessmodell ist der zweite Beitrag dieser Arbeit. Der dritte Beitrag ist eine Erweiterung eines bestehenden, auf Aggregation basierenden Autorenprozesses. Solche Autorenprozesse umfassen bislang nur die reine Inhaltsproduktion. Als Erweiterung wurde dem ausgewählten Autorenprozess eine Designphase zur didaktischen Planung hinzugefügt. Darüber hinaus werden aus der Aggregation entstehende Kontextinformationen genutzt um die Beschaffung von integrierbaren Lernressourcen zu verbessern. Hat man auf diese Weise Lernressourcen aus verschiedenen Quellen aggregiert, so erscheint die entstandene Lernressource oftmals uneinheitlich. Es ist daher nötig, eine Anpassung der aggregierten Lernressourcen vorzunehmen um ein einheitliches Erscheinungsbild zu erreichen. Diese Arbeit definiert ein Framework für die Darstellung und Anpassung der Inhalte von Lernressourcen. Dieses Framework ermöglicht die Entwicklung von Anpassungswerkzeugen, die unabhängig von spezifischen Dokumentenformaten arbeiten. Eine Lernressource kann hierfür ganzheitlich betrachtet und bearbeitet werden, anstatt nur in Form einzelner, enthaltener Dokumente. Und schließlich bietet der fünfte Beitrag der vorliegenden Arbeit einen neuen Ansatz für die thematische Klassifikation von Lernressourcen. In Fällen, wo kein geeigneter Trainingskorpus für die Nutzung von Methoden des maschinellen Lernens verfügbar ist, wird die Internet-Enzyklopädie Wikipedia als Ersatzkorpus vorgeschlagen. Diese Dissertation zeigt eine konkrete Implementierung des vorgeschlagenen Ansatzes auf, sowie eine Kalibrierung von Parametern für die genutzte Methode. Eine Evaluierung des so erstellten Klassifizierers zeigt, dass der gewählte Ansatz unter den Bedingungen spärlicher Trainingsdaten signifikant bessere Ergebnisse erzielt als herkömmliche Ansätze.

## Abstract

Since computers became omnipresent in working environments and households, people have also started to use them for education. The term *e-learning* (electronic learning) refers to scenarios whereby learning or teaching is assisted by computers. On the one hand, e-learning may support efficiency of teaching; on the other hand, learners gain the freedom to learn when, where and how they want. Learning resources comprise of digital materials – e. g. documents, images, videos, simulations, assessments, etc. – used in educational scenarios. When learning takes place in a formal way, a teacher usually has to provide these learning resources to the learners.

The production of learning resources either by teachers themselves or by third party content authors has been investigated by many researchers. It is widely accepted that the production of high quality content is labor intensive work that incurs high costs. Therefore, it is advantageous to reuse existing learning resources in order to reduce costs. Besides using learning resources repeatedly for other courses, learning resources may also be seen as the preliminary products of the authoring process. Similar to the reuse of software libraries in software development, the reuse of learning resources is said to reduce the production costs of new learning resources. As e-learning is a popular medium for educational scenarios in academia, vocational training, and product training, today the reuse of learning resources remains relevant.

This thesis in particular considers the scenario of reuse in which existing learning resources serve as preliminary products for the creation of new learning resources for Web based training. Authors are interested in reusing the learning resources created by other authors. It is assumed that these authors belong to different organizations. Furthermore, these authors do not use a common authoring tool because they are obliged to use the tools specified by their respective organizations. There are content models which specify how learning resources may be constructed hierarchically. Authoring paradigms, such as authoring by aggregation, allow in principle a new learning resource to be created as the aggregation of different smaller learning resources. However, it is necessary that the learning resources to be combined are stored as individual resources. This approach works well if an organization systematically creates fine-grained, modular learning resources by using a suitable authoring environment. Many authoring tools use arbitrary content formats that are incompatible with other authoring tools or learning management systems. Thus, learning resources are not exchanged in their source format; instead, the Shareable Content Object Reference Model (SCORM) specifies a common exchange format for the learning resources. Learning resources are exported into the SCORM format for exchange. One disadvantage of this is that the modular components of a learning resource are no longer able to be distinguished as individual learning resources.

This thesis enables the reuse of modular learning resources, which have due to an export process ceased to exist as individual learning resources. Two factors shape the foundation of this thesis: firstly modularity with respect to learning resources and learning resource specifications; and secondly the multi-granularity reusability of learning resources. The definition of modularity requires that parts of

a learning resource may be encapsulated, exposed and reused separately as modules. Multi-granularity reusability extends the requirements of modularity to ensure that *each relevant part* of a learning resource can be reused modularly.

In the pursuit of multi-granularity reusability, contributions within this thesis have been aligned with six criteria in total. These criteria include on the one hand, the three technical requirements of multi-granularity reusability – *availability*, *retrievability* and *interoperability* – and on the other hand, the three dynamic modular operations required for the reuse of modular learning resources: *modularization*, *aggregation* and *adaptation*. There are five contributions within this thesis which each cope with either one or more of these six criteria. All in all, these five contributions bring the reusability of learning resources closer towards the goal of multi-granularity reusability.

In the first contribution, an extension to the SCORM specification has been defined which enables the modular reuse of parts of a SCORM package and allows these learning resources to be modularized and aggregated. Furthermore, several approaches for modularization have been reviewed. As a result, a generic process model for the modularization of learning resources resulted from these various approaches. This process model is the second contribution of this thesis. The third contribution is an extension of an authoring by aggregation process. The authoring by aggregation within existing implementations is restricted to pure content development only. This thesis has extended one of theses processes by a design phase which integrates the light-weight authoring approach of authoring by aggregation. In addition, context information from the aggregation process is utilized to improve the retrieval of learning resources for aggregation. After learning resources from different origins have been obtained and aggregated, the aggregation often looks like a patchwork. It is necessary to adapt the aggregated learning resources towards a unified appearance. This thesis proposes a framework for learning resource content representation and adaptation. This framework enables the development of adaptation tools which are able to work independent of different document formats and focus on a learning resource in its entirety instead of on individual documents. Finally, the fifth contribution in this thesis is a new approach for the topical classification of learning resources. For cases in which no suitable training corpus is available, Wikipedia the online encyclopedia is used as a substitute corpus for training machine learning classifiers. This thesis provides an actual implementation of the proposed approach using the k-nearest-neighbor method and the calibration of suitable parameters. An evaluation of the Wikipedia-based classifier has shown that it performs significantly better than traditional approaches even if only a few sample learning resources are available for training.

## Acknowledgments

The present thesis was created while I was working in parallel for the Multimedia Communications Lab (KOM) at the Technische Universität Darmstadt and for the SAP Research CEC Darmstadt. I want to thank both for their support of my research that led to this dissertation.

I would like to thank my doctoral supervisor Prof. Dr.-Ing. Ralf Steinmetz for giving me the opportunity to write a dissertation at KOM. It was a pleasure to work in this great environment that has inspired me for conducting research. Thanks also go to my second advisor Prof. Dr.-Ing. Abdulmotaleb El Saddik, who has provided valuable feedback to my research.

Within KOM I worked in the knowledge media group led by Dr.-Ing. Christoph Rensing. Special thanks go to Christoph for his guidance and support. I would also like to thank the rest of the group for countless discussions and suggestions that have contributed to the success. Particularly I want to mention the Content Sharing team: Birgit, Tomas, and Sonja. I also thank all students who have conducted their bachelor or diploma theses with me.

Thanks also go to SAP Research and the director of the CEC Darmstadt, Dr. Knut Manske. SAP has given me the chance to conduct research from a business-oriented perspective. I would also like to thank Dr.-Ing. Andreas Faatz, who has helped me several times to put my ideas and thoughts in order.

I am also grateful to all those colleagues who have proofread my dissertation or parts of it. These are particularly Andreas, Birgit, Matthias, Philipp, Stefan, and Tobias. Last but not least I want to thank my parents who have always supported me.

## Contents

# Part I

# Towards Modularity and Multi-Granularity Reuse of Learning Resources

# 1 Introduction

## 1.1 Motivation

Since computers became omnipresent in working environments and households, people have also started to use them for education. The term *e-learning* (electronic learning) refers to scenarios whereby learning or teaching is assisted by computers. On the one hand, e-learning may support efficiency of teaching; on the other hand, learners gain the freedom to learn when, where and how they want. Learning resources comprise of digital materials – e. g. documents, images, videos, simulations, assessments, etc. – used in educational scenarios. When learning takes place in a formal way, a teacher usually has to provide these learning resources to the learners.

The production of learning resources either by teachers themselves or by third party content authors has been investigated by many researchers. It is widely accepted that the production of high quality content is labor intensive work that incurs high costs. Therefore, it is advantageous to reuse existing learning resources in order to reduce costs. Besides using learning resources repeatedly for other courses, learning resources may also be seen as the preliminary products of the authoring process [Hod02a]. Similar to the reuse of software libraries in software development, the reuse of learning resources is said to reduce the production costs of new learning resources. As e-learning is a popular medium for educational scenarios in academia, vocational training, and product training, today the reuse of learning resources remains relevant.

This thesis in particular considers the scenario of reuse in which existing learning resources serve as preliminary products for the creation of new learning resources for Web based training (WBT). Authors are interested in reusing the learning resources created by other authors. It is assumed that these authors belong to different organizations. Furthermore, these authors do not use a common authoring tool because they are obliged to use the tools specified by their respective organizations. There are content models which specify how learning resources may be constructed hierarchically. Authoring paradigms, such as authoring by aggregation [Hoe05], allow in principle a new learning resource to be created as the aggregation of different smaller learning resources. However, it is necessary that the learning resources to be combined are stored as individual resources. This approach works well if an organization systematically creates fine-grained, modular learning resources by using a suitable authoring environment [BLW99]. Many authoring tools use arbitrary content formats that are incompatible with other authoring tools or learning management systems. Thus, learning resources are not exchanged in their source format; instead, the Shareable Content Object Reference Model (SCORM) [Adv] specifies a common exchange format for the learning resources. Learning resources are exported into the SCORM format for exchange. One disadvantage of this is that the modular components of a learning resource are no longer able to be distinguished as individual learning resources.

The goal of this thesis is to enable the reuse of modular learning resources which have due to an export process ceased to exist as individual learning resources. These learning resources are made available once again as preliminary products for the creation of new learning resources.

## 1.2 Goals

Existing authoring systems enable the reuse of individual learning resources. Hierarchical content models are used as the basis for creating and storing fine-grained learning resources which can be combined to form larger learning resources. Thus, the reuse of learning resources is made possible at multiple levels of granularity. When learning resources are exchanged between organizations, transformation into the SCORM format prevents further separate reuse of fine-grained learning resources.

This thesis focuses on the multi-granularity reusability of learning resources without restrictions due to a particular authoring tool's format. Multi-granularity reusability is understood as the ability to separately reuse parts of a learning resource's content. Users should have a choice of the parts which they want to reuse. Multi-granularity reusability requires that parts of a learning resource with the potential to be reused remain available and retrievable for reuse, and that they are interoperable so that they can be aggregated to new learning resources.

## 1.3 Contributions

In the pursuit of multi-granularity reusability, contributions in this thesis align with six criteria in total. These criteria are on the one hand three technical requirements of multi-granularity reusability – *availability*, *retrievability* and *interoperability* – and on the other hand three dynamic modular operations that are required for the reuse of modular learning resources: *modularization*, *aggregation* and *adaptation*. There are five contributions within this thesis which each cope with one or more of these six criteria. All in all, the five contributions work towards the goal of multi-granularity reusability.

- Multi-granularity reusability among heterogeneous systems is achieved through the de facto standard SCORM. SCORM does not fully support the modularity of learning resources. Therefore, the first contribution of this thesis is the extension of SCORM with a module concept, which combines the advantages of a widespread standardized learning resource format with those of modularity. This module concept serves as the foundation of the remaining contributions.

- Having a learning resource format specification which supports modularity is an essential foundation of multi-granularity reusability. However, it also requires that learning resources are created which utilize the modularity prospects gained. Assuming that learning resources exist as traditional SCORM packages without any modularity support, it is a challenge to transform such monolithic learning resources into a modular form. This process of transforming a learning resource into an aggregation of smaller learning resources is called *modularization*. This thesis puts forward a reference process model for the modularization process which has been derived after a review of existing modularization approaches.

- Another issue regarding multi-granularity reusability is how to aggregate learning resources into larger structures. There is an authoring paradigm called *authoring by aggregation* which centers the creation of learning resources upon aggregation. Existing authoring by aggregation processes are extended by a separate phase for didactic design. Furthermore, the retrieval of learning resources from existing repositories is seen as an integral part of authoring by aggregation. The usage of aggregation context (knowledge about other learning resources within an aggregation) has been proposed to improve the retrieval of learning resources.

- When learning resources from different origins are aggregated a so-called *mosaic effect* may occur: the aggregation looks like a patchwork, because the layout, design, writing styles and pedagogical styles are different. Often an adaptation is required in order to achieve consistency. Adapting learning resources which consist of several documents with possibly different document formats for consistency is challenging. This thesis suggests a framework for the abstraction of content representation and adaptation. The framework provides a single interface for the analysis and modification of a learning resource including all relevant contained documents.

- Finally, the retrievability of learning resources is improved by the development of a new topical metadata generation method. Learning resources are categorized according to the subject covered. While traditional machine learning based categorization methods need a training corpus of manually tagged learning resources, the new approach uses articles from Wikipedia[1] for training. This thesis proposes a classifier which categorizes learning resources into the Wikipedia category system. Optimal parameters for such a classifier have been experimentally determined.

## 1.4 Organization

This thesis is divided into four parts. The first part consists of this introduction and a review of related work in Chapter 2. Related work on learning resources and reusability is reviewed and discussed in this chapter. The chapter ends with definitions of modularity and multi-granularity reusability. One aspect of modularity is that modular operations are available. Part II consists of three chapters which cover three non-trivial modular operations: modularization, aggregation and adaptation. Chapter 3 introduces a module concept based on SCORM and a reference model for the modularization process. The aggregation of learning resources and retrieval of learning resources for aggregation is covered in Chapter 4. In this chapter, an existing authoring by aggregation process is extended in order to support additional phases of content development. In Chapter 5, a framework for content representation and adaptation is presented. This framework provides an abstraction layer for the implementation of adaptation tools. The third part of this thesis focuses on the improvement of retrievability of learning resources by the automatic generation of metadata. Chapter 6 introduces and evaluates a new method for Wikipedia-based metadata generation. Wikipedia replaces traditional training corpora; thus the classifier can be applied in situations in which traditional corpora do not exist. After that, Part IV completes the thesis with a proof-of-concept implementation and conclusions. An implementation of the concepts from Part II as a prototype is described in Chapter 7. The prototype is evaluated with respect to the requirements

[1] Wikipedia is a free online encyclopedia: http://www.wikipedia.org

of modularity and multi-granularity reusability. Finally, Chapter 8 summarizes this thesis and provides an outlook on future research issues.

# 2 Reuse of Learning Resources

The focus of the thesis is to improve reusability of learning resources, particularly in heterogeneous systems. To introduce the topic, this chapter discusses the state of the art of learning resources and reusability in Section 2.1. Definitions of modularity and multi-granularity reusability are derived in Section 2.2. Section 2.3 provides a use case example and a scenario of multi-granularity reuse in heterogeneous systems. Challenges for multi-granularity reusability are derived from this scenario; And finally, building blocks are identified that contribute to an overall improvement of multi-granularity reusability.

## 2.1 Related Work

This section discusses the related work about reuse in e-learning. Starting with e-learning in general the section moves on to more specific topics about reuse. Different definitions of learning resources are compared, as well as content models and learning object repositories.

### 2.1.1 E-Learning

Computers and related technology have been used for a long time for educational purposes. The term e-learning (or electronic learning) stands for computer-supported learning. Thus, e-learning embraces face-to-face learning[2] as well as distance learning, formal and informal learning, various pedagogical approaches and technologies. Another term, which is often used synonymously is *Technology Enhanced Learning*.

Distance learning especially benefits from electronic support, as it is now much easier to transfer learning materials to a distant learner, and to enable him to interact with teachers and other learners. Various technologies are used in the area of distance learning, for instance computer-based trainings, Web based trainings (WBT), capturing and replay of activities [SS03a], or tools for Computer Supported Collaborative Learning [HSW04]. Computer-based trainings are special computer programs for an educational purpose. Typically, computer-based trainings are stand-alone computer programs, which run locally on a learner's computer. With the spreading of broadband internet access, Web based trainings have become popular. Web based trainings are similar to computer-based trainings, but are accessed by the learner via an internet browser.

Web based trainings are typically made available to learners by learning management systems (LMS). A LMS not only provides WBTs as a specialized form of learning resources, but may also track the learning progress and brings together communities of learners with similar interests.

[2] For instance interactive lectures [KE07] and digital lecture halls [MT02]

### 2.1.2 Reusable Learning Resources

Because the production of learning resources may be quite expensive, there is a demand for reusing existing learning resources in order to decrease the costs of teaching. In the course of standardization of reuse in the e-learning field, the need for definitions of (digital) learning resources became clear. During the last ten years, many researchers and organizations have tried to define reusable learning resources. The IEEE Learning Technology Standards Committee (LTSC) has defined a learning object (LO) very broadly as "any entity – digital or non-digital – that may be used for learning, education or training" [Hod02b].

Wiley discussed in 2000 several learning object definitions that were available at that time [Wil00]. He states that the concept of reusable learning objects is based on the paradigm of object orientation from software engineering. Wiley especially emphasizes the reuse of learning objects:

> "This is the fundamental idea behind learning objects: instructional designers can build small (relative to the size of an entire course) instructional components that can be reused a number of times in different learning contexts." [Wil00]

Furthermore, Wiley identifies the novelty of reusable learning objects:

> "(...) any number of people can access and use them simultaneously (as opposed to traditional instructional media, such as an overhead or video tape, which can only exist in one place at a time). Moreover, those who incorporate learning objects can collaborate on and benefit immediately from new versions. These are significant differences between learning objects and other instructional media that have existed previously."

Criticizing the LTSC's too broad definition of learning objects, Wiley limits a learning object to "any digital resource that can be reused to support learning." This definition stresses on the one hand that only digital resources are meant, and on the other hand that only reusable resources are of interest. In his paper Wiley also discusses the issue of granularity of learning objects – which scope should a learning object have? He postulates that learning objects of different complexity may exist, and also defines an exemplary taxonomy.

Similarly to Wiley, Hodgins recognized the economic dimension of reuse [Hod02a]. Analogue to the introduction of reusable components in industrial manufacturing, Hodgins expects economies of scale in the reuse of learning objects. In [Hod02a] he describes the Autodesk Content Hierarchy as an example for a multi level content hierarchy of learning objects. The model consists of several levels of increasing complexity. In a thought experiment Hodgins imagines how a perfect scenario of reuse would look like. Considering a future, when millions of learning objects are available for reuse, Hodgins identifies metadata as an important element. Without adequate metadata, suitable reusable learning objects are unlikely to be found. The necessity for retrieval rises with the number of existing learning objects.

In 2003, Polsani has reviewed the existing definitions of learning objects and names three common attributes [Pol03]:

- *Accessibility:* "the LO should be tagged with metadata so that it can be stored and referenced in a database"

- *Reusability:* "once created, a LO should function in different instructional contexts"
- *Interoperability:* "the LO should be independent of both the delivery media and knowledge management systems"

As Hodgins before, Polsani also regards learning objects as reusable components, which have to be reusable in different contexts. His understanding of a reusable learning object is based on two principles: *learning* and *reusability*. In contrast to former concepts, Polsani demands that learning objects are dedicated to the purpose of learning. Therefore, he gives a new definition of a reusable learning object as "an independent and self-standing unit of learning content that is predisposed to reuse in multiple instructional contexts." [Pol03] Polsani also discusses the issue of granularity; he recommends to create learning objects, which cover exactly one topic. Learning objects are furthermore considered to be composed of smaller elements, such as text, images, or video.

At the same time, Koper narrows Wiley's definition into another direction:

> "I will further narrow down the scope of this definition for this chapter to: 'any digital, reproducible and addressable resource used to perform learning activities or learning support activities, made available for others to use.'" [Kop03]

Koper demands of learning objects to be addressable and accessible by other users. His definition is based on the idea that resources, which are accessible via the Internet, provide best reusability. Having the point of view of a teacher, Koper considers learning objects not only for distance education. He also regards presence scenarios by defining *units of learning*. A unit of learning – in contrast to a learning object – contains also learning activities. Learning activities are any actions learners or other involved roles perform for the purpose of learning. These activities often include the use of digital learning resources.

Duval and Hodgins have published a research agenda on learning objects and metadata [DH03]. This agenda adopts the learning object definition of the LTSC and the Autodesk LO taxonomy consisting of five different granularities; these granularities are *data or raw media elements*, *information objects*, *application objects*, *aggregation assemblies*, and *collections* (see Figure 2.1, cf. [Hod02a]). The main focus of that paper, though, is on metadata, its creation and utilization for retrieval, aggregation, and usage of learning objects. Three of the listed open research issues are metadata generation, authoring by aggregation (creation of a LO by composing smaller learning objects) and decomposition of existing LOs into smaller learning objects. These research issues are dealt with in more detail in subsequent chapters of this thesis.

A formalized usage of metadata is described by Sicilia and Sanchez-Alonso [SS03b]. They transfer the concept of *design by contract* from object-oriented software engineering to the specification of learning objects. Each learning object is specified by a contract, which consists of pre- and postconditions. The requirements of a learner and his context are the preconditions under which the learner may use the learning object. The postconditions express the ensured learning outcome, i. e. the learner's knowledge. The approach of Sicilia and Sanchez-Alonso is based on Polsani's LO definition; a learning object is regarded as a component, which transforms a learner from one state into another (typically his knowledge is increased).

There are also a lot of other attempts to deal with reusability. They differ in the considered education scenarios and in which kinds of learning objects are supported. As most of the discussed approaches are focused mainly on static, document-like learning objects, another work on reusable multimedia learning

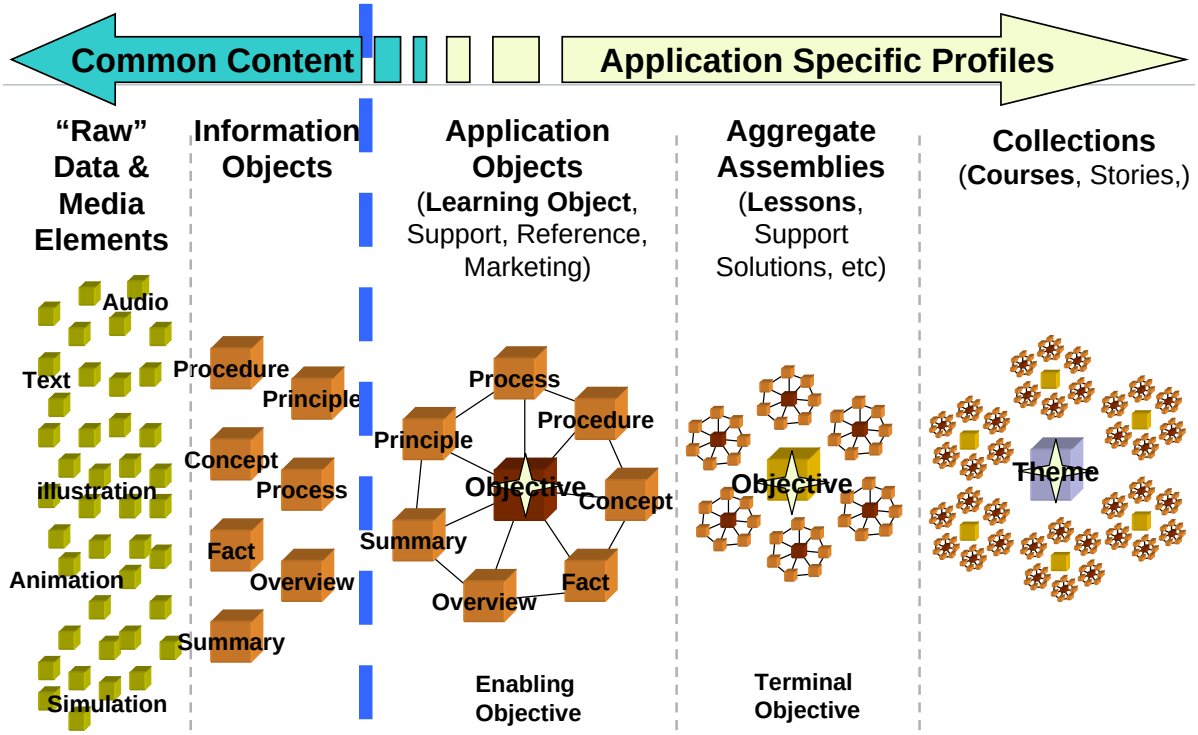


Figure 2.1: Content hierarchy of the Learnativity Content Model [DH03].

objects should be mentioned to illustrate the variety of approaches. El Saddik et al. have specified smart dynamic multimedia learning objects [ESFS01]. Using metadata as an interface description, several of these smart learning objects can be composed in order to interact with each other.

### 2.1.3 Content Models and Instructional Design

Many issues regarding learning objects have a technical dimension, as the previous sections have shown. However, the actual purpose of learning objects is education. Learning objects are used to teach and to learn. Therefore, pedagogy – or didactics – is an important aspect of a learning object. With the emergence of e-learning, new styles of teaching have evolved. Pedagogy is based on learning theories; a learning theory is a psychological explanation for how learning is assumed to happen within a person. Different learning theories, such as behaviorism, cognitivism or constructivism, exist and partly contradict each other [NHHM+04]. These learning theories also imply different instructional theories. In English, the term *instructional design* is commonly used to describe the task of arranging contents and communication for educational use. According to Reigeluth, instructional design "is concerned with producing knowledge about optimal 'blueprints' – knowledge about what methods of instruction will optimize different outcomes" [Rei83]. Instructional design theory should the basis for the creation of learning resources [Wil00]. The IMS Learning Design specification provides a framework for modeling instructional design, especially with regard to e-learning applications [KOA03]. For the German e-learning community, Meder has developed the so called *Web didactics* (German: "Web-Didaktik") out of the traditional European general didactics [Med06]. This Web didactics can be seen as a guideline for systematic creation of e-learning contents and processes.

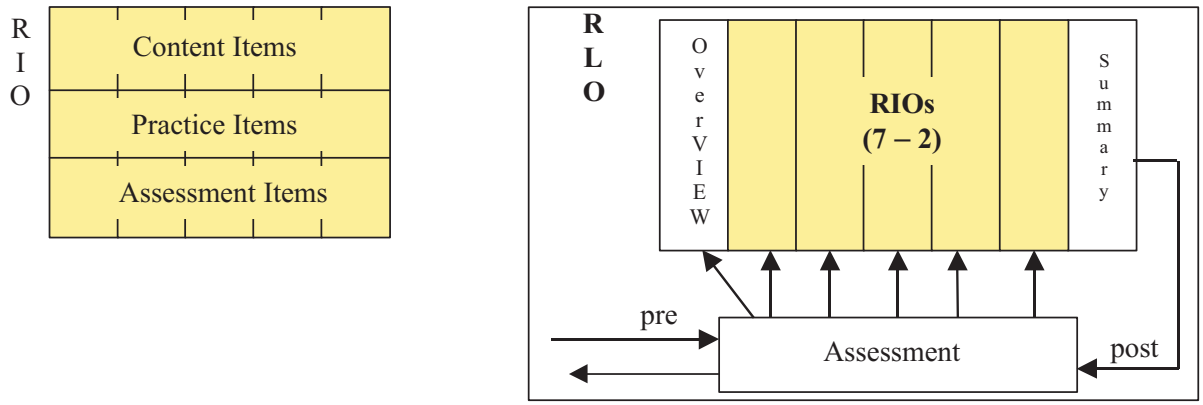


Figure 2.2: Cisco Model – RIOs and RLOs [BLW99]

Didactics are not only useful for classroom scenarios. WBTs also require didactics to enable successful learning. Basically, each teacher may plan and realize his WBT freely with regard to structure, contents, media usage and application of interaction and communications. In practice, though, content models have evolved. These content models are templates for structure and didactic arrangement of (mostly passive) Web based trainings. Usually, a content model specifies different levels of granularity of learning objects and how multiple objects of one granularity are aggregated on the next higher granularity level.

Content models have been developed in the first place by companies that produce large amounts of learning objects. For them, the standardization of a learning object's structure and sequencing leads to better quality and reduced costs.

One of the content models – the Learnativity Content Model has already been mentioned above (Figure 2.1). The Learnativity Content Model specifies a taxonomy of five granularity levels ranging from raw media elements to whole curricula. Each granularity level is an aggregation of elements of the preceding level. The Learnativity Content Model only describes the educational granularity of the contents, but is not a technical specification.

Cisco Systems has defined its own content model for their training materials [BLW99]. The Cisco content model defines two granularity levels: Reusable Information Object (RIO) and Reusable Learning Object (RLO). Additionally, smaller elements (comparable to the raw media elements of the Learnativity Content Model) are mentioned in the authoring guidelines. A RIO has a single learning objective and contains content items, practice items and assessment items. There are five different types of RIOs: concept, fact, procedure, principle, and process. A RLO is a complete lesson, consisting of 5 to 9 RIOs plus an overview, a summary, and an assessment (Figure 2.2). Cisco provides templates, guidelines and an authoring tool for authors of RLOs and RIOs. After creation, the contents may be packaged for different delivery channels, such as dynamic Web packages, CD-ROMs, or instructor-led training materials.

The Advanced Distance Learning Initiative (ADL) has been founded by the US Department of Defense in order to improve and standardize tools for Web based distance learning. ADL has assembled the Sharable Content Object Reference Model (SCORM) out of different existing specifications and standards, such as IEEE Learning Oject Metatada (LOM) or IMS Content Packaging. SCORM is today the defacto standard for exchanging WBT contents, as almost every learning management system is SCORM compliant. In the current version SCORM 2004, the reference model is composed of three books: the Content Aggregation Model (CAM), Sequencing and Navigation (SN) and the Run-Time Environment

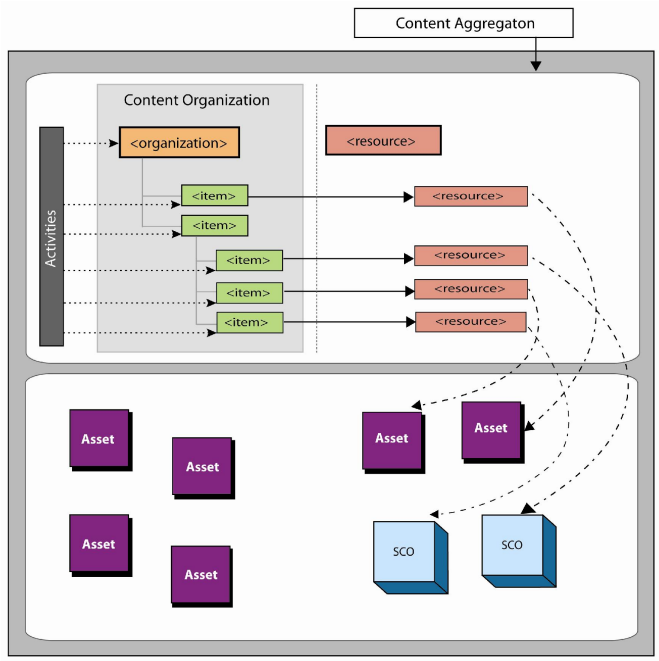


Figure 2.3: Content aggregation in SCORM [Adv]

(RTE) [Adv]. The RTE standardizes the communication between the LMS and the contents, such as session management or data transfer. For instance, the result of an assessment can be transmitted to the LMS for persistent storage via a common interface. Sequencing and Navigation is used to define the order in which contents are presented to the user.

The most relevant book of the SCORM specification for this thesis is the Content Aggregation Model, as it specifies "assembling, labeling and packaging of learning content". According to the CAM, SCORM compliant contents are bundled as a package; this package contains the content files and a manifest file, describing structure and metadata of the contents. The content model allows the following structural elements: assets, Sharable Content Objects (SCO), activities, content aggregations and organizations. Assets are the smallest components: any media that can be rendered in a Web browser, such as text, images, videos or assessments. Assets can also be assembled from other assets. A SCO is either a single asset or a collection of assets, complying with a particular interface. Through this interface the SCO communicates with the RTE; for instance, the learning progress for a SCO can be tracked by the learning management system. There is no restriction of the pedagogical complexity for a SCO.

In order to build larger units, activities may be specified; an activity is either the execution of one SCO, or an aggregation of other activities. The sequence of these activities for learning is either given by the tree-like aggregation structure, or by more complex Sequencing and Navigation instructions. The big picture of content aggregation in SCORM is illustrated in Figure 2.3.

All of the content models presented in this section have in common that they define multiple levels of granularity, which are respectively aggregated to larger units. The properties of these units at different levels of granularity vary. Some unit sizes are only raw media without didactic intention, others have dedicated didactic functions; larger units again do not have didactic functions but are self-contained and intended to achieve a given learning objective. A detailed comparison of different content models has been compiled by Verbert and Duval [VD04].

It appears that content models regard an aggregation of content units only as a target of a reuse process, but not as a source of learning resources for reuse. Learning resources for reuse are obtained in the appropriate granularity from a repository. It is not intended to break an aggregation into its constituting learning resources in order to reuse them separately. Thus, existing content models support reuse of learning resources that each may have *one out of multiple specified levels of granularity* and to aggregate them to higher levels of granularity.

Content models specify the structure of learning resources and particularly which elements may be aggregated in which order with other elements. The structure of learning resources is well specified and also standardized for exchange between different tools (i. e. SCORM), whereas the content format of assets is not standardized. Assets within SCORM packages are often delivered as HyperText Markup Language (HTML) documents. Some authoring tools, such as LearnCube[3], ship arbitrary content players along with each SCORM package (in the case of LearnCube the player is developed as an Adobe Flash[4] file). The variety of content formats makes reuse difficult, especially if changes of the content are required. There are efforts for specifying XML[5]-based content formats specifically for e-learning content. An example for such initiatives is the Learning Material Markup Language Framework [SF02]. However, there is no specific e-learning content format that is widely spread and accepted as de facto standard. Thus, even if content models are harmonized, the incompatibility of content formats is still a challenge. Recently, a new project called *Resource Aggregation Model for Learning, Education and Training* (RAMLET)[6] was initiated by the IEEE Learning Technology Standards Committee. This project is going to specify a common aggregation model, which can be mapped to different existing content models.

### 2.1.4 Learning Object Repositories

For effective reusability it is not enough that learning resources are reusable in an educational and technological sense (that is, a learning resource can be used by a user, and the learner successfully learns from the usage). They also have to be accessible, as already mentioned in the previous section. Thus, reusability is not only a property of learning resources, but also a requirement towards supporting systems. There are several types of systems, which are involved over the life cycle of a learning resource: authoring environments during the creation phase, storage systems for storage and distribution, and finally learning management systems for learning.

If learning resources are to be reused, they have to be stored and made accessible. Databases for learning resources are called learning object repositories (LOR). A learning object repository is a digital archive for learning resources, which enables users to upload, search and download learning resources. Thus, a LOR is a special case of a content management system[7]. Many different variations of LORs exist. Some repositories are completely open to everybody, such as MERLOT [8]. Others, such as the Ariadne repository [DFC+01], are available only to a closed community, where members pay for the access. The Content Sharing project [The] has worked towards a commercial learning resource marketplace, where

[3] http://www.x-pulse.de/learncube.php
[4] http://www.adobe.com/support/documentation/de/flash/
[5] Extensible Markup Language
[6] http://www.ieeeltsc.org/working-groups/wg11CMI/ramlet
[7] Definitions, system architectures and implementation details for content management systems can be found in [MT04].
[8] http://www.merlot.org

access to the system is open for everyone, but users have to pay for downloading and using learning resources. A study in 2002 has revealed that the size of LORs varied between two-digit numbers and up to 15,000 learning objects [ND02]. Today, more than five years later, probably even larger repositories exist.

Learning object repositories also differ in which learning resource formats they support. Some repositories are restricted to a selected number of formats; others (e. g. MERLOT) allow almost everything, even Web links. ResourceCenter [HHRS05], for instance, supports only its own proprietary format, but integrates an authoring environment with the repository. The IMS Global Learning Consortium has specified[9] a set of core functionality a repository should provide.

Storing learning resources as files in a repository is not enough. They have to be retrievable, and users should be able to get a quick overview over the contents. Metadata is used to summarize the contents and other attributes of a learning resource. In the past, there have been several approaches to specify metadata formats [Ste01]. Finally, the Learning Object Metadata (LOM) specification has prevailed. It has also been accepted as an IEEE standard [Hod02b]. However, LOM has some major drawbacks: insufficient interoperability and a conflict of goals between automatic and human processing. LOM has been designed for enabling interoperability - but only on a syntactical level. The semantics of LOM entries, though, are not fully standardized. Two LOM-compliant applications may both read and write LOM records, but they cannot necessarily 'understand' the attribute values written by the other application [SS04, BPN03]. The second drawback is that LOM is used for both automatic processing and presentation to humans. Feedback from users in the Content Sharing project[10] indicated that some fields are too ambiguous to be understood by users. Other fields are unspecific free text fields, making it hard to process the values by algorithms [KdHWA04, SGP+05].

Assuming that learning resources can be found within a repository by its metadata, another challenge arises. There is a large number of repositories. How can a desired learning resource be found, which might be located in any of these repositories? The CORDRA project [RDL05] currently aims at developing a solution for a global federation of LORs. As long as a global federation is not available, some intermediate solutions are applied. For instance, Global Learning Objects Brokered Exchange (GLOBE) is a federation of five well-known repositories, such as MERLOT and Ariadne. They have specified a common query interface for inter-repository queries, called Simple Query Interface (SQI) [SMvA+05]. Meanwhile, a number of projects have adopted SQI. SQI is only a query interface, but not a query language; it can be used for session management and for the transmission of queries and result sets. For the actual query, another specification is required. Most SQI-compatible repositories use the Very Simple Query Language (VSQL) as their query language. VSQL allows only to search by a list of keywords which are matched against some or all LOM entries for a particular learning resource. It is not possible to search for specific LOM fields. The ProLearn project develops a new, more powerful query language called ProLearn Query Language (PLQL) [CCD+07].

Besides central repositories, some peer-to-peer systems for storage and distribution of learning objects have been developed, such as Edutella and LOMster [NWQ+02, TDV02]. In these systems, the central repository is replaced by a large number of peers, which contribute their learning objects to the com-

[9] IMS Digital Repositories Interoperability - Core Functions Information Model: http://www.imsglobal.org/digitalrepositories/driv1p0/imsdri_infov1p0.html

[10] http://www.contentsharing.com

munity. Naturally, peer-to-peer systems focus more on sharing learning objects than on archiving them persistently.

### 2.1.5 Reuse and Granularity in Other Areas

E-learning is not the first application area in which reuse plays an important role. Industrial manufacturing has benefitted for a long time from standardized parts that can be assembled to different products. Examples are the assembly of cars or personal computers, where most parts are obtained from suppliers. However, this is a lopsided comparison, as the parts to assemble have to be physically built in order to be used.

More similar to reuse in e-learning is code reuse in software development. Code reuse is a valuable comparison particularly because it demonstrates the variety of what can be understood as reuse. First of all, simply copying lines of code from one computer program to another one is already code reuse. And this method is also likewise applied for authoring of e-learning contents. But software engineering soon advanced to encapsulation of code to self-contained components, so that portions of code could be reused several times without copy&paste. There are a lot of different approaches of encapsulation, leading also to various forms of reusable code, called *methods*, *classes*, *libraries*, or *components* [Mey97]. Currently, the concept of reusable *Web services* is en vogue [Alo04, Ber08]. Nevertheless, the basic principle remains always the same: A portion of code is encapsulated and made available via an interface.

Though there are many parallels between reusable software components and reusable learning objects, there are also differences. While software components manipulate mainly data, hiding the particular implementation from the developer and user, learning objects interact with the learner. In consequence, the implementation – that is, the contents of a learning object – is always visible to the user. This fact will be important, when aggregation is discussed later.

Reusable software components can be aggregated. Methods may call other methods, classes may use other classes, and Web services may be orchestrated. As mentioned above, the aggregation of software code relies mainly on interfaces, hiding the actual implementation. A developer usually can infer from the interface whether and how to use a software component. Sometimes, also the context and behavior of a component may be specified (e. g. the Hoare logic [Hoa69] or Design by Contract [Mey97]). By aggregation of software components, larger software components are created. In some cases, dedicated types of components allow a distinction of granularity. But more often, granularity or complexity is not explicitly distinguished (e. g. the composition of multiple Web services is again a Web service).

Another domain, which involves multiple levels of granularity is multimedia retrieval. Multimedia archives store multimedia objects of different complexity. When a user searches for multimedia contents, his query often does not address the whole multimedia object, as it is stored in the database, but possibly only parts of it - for example only a particular frame or scene of a movie is of interest. Multimedia retrieval systems cope with this granularity issue by segmentation of contents at different levels of granularity; the segments are made individually retrievable [Sch05]. The same applies to the retrieval of e-books: the user does not always want to retrieve a whole book, which contains his search terms spread over all pages, but rather only few pages or a chapter [HTZH07].

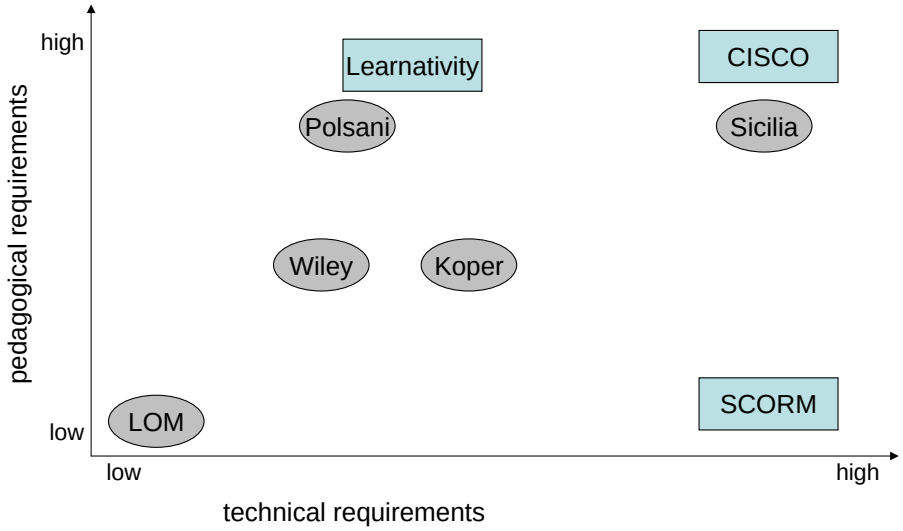


Figure 2.4: Classification of learning object definitions and content models.

## 2.2 Review of Existing Approaches and Derived Definitions of Reusability

As Section 2.1 has pointed out, reuse in e-learning involves different concepts: learning objects, reusability, LORs, metadata, content models, and Web didactics. Unfortunately, for most of these concepts quite different definitions and embodiments exists. This section resumes the observed definitions from the previous section, tries to extract a consensus and derives definitions valid for this thesis.

The definitions of learning objects and learning resources should be reviewed with respect to content models: Some definitions of a learning object relate to a particular pedagogical granularity within a given (or implicit) content model [Hod02a]; other definitions consider *learning object* as an embracing term for all entities of a content model [Hod02b].

### 2.2.1 Notions of Reusability

In fact, the requirements on learning objects from the various definitions can be classified in two dimensions: a pedagogical and a technical dimension. Pedagogical requirements are, for instance, Polsani's demands for independence and the dedication to the purpose of learning, and the approach of Sicilia and Sanchez-Alonso that learning objects perform a measurable transformation of a learner's state of knowledge. Technical requirements are e. g. accessibility and interoperability. Figure 2.4 illustrates what requirements the various definitions of learning objects and content models contain with respect to the technical and pedagogical dimensions. Interestingly, the most successful standards – LOM and SCORM – do no make pedagogical demands on learning objects. Based on this finding, a broad definition without strict pedagogical limitations will be used throughout this thesis. As a working definition, we define the term learning resource (LR) as:

**Definition 2.1 (learning resource)**
*A **learning resource** is a digital resource used for e-learning [RBH+05].*

The term *learning resource* instead of *learning object* has been chosen, because it emphasizes the resource character of our concept. Learning objects on the other hand are often associated with either pedagogical properties or specific technical interfaces. In the remainder of this thesis, every digital resource used for learning is considered to be a learning resource. Where a narrower definition is needed, it is specified in the respective chapter.

Tightly connected to learning object definitions is the notion of *reuse* and *reusability*. Again, there are a pedagogical and a technical dimension of reusability. Furthermore, reusability can be understood either as a property of a learning object, or as functionality of systems that deal with learning objects. From a pedagogical point of view, reusability principally means that a learning object, which is pedagogically useful for its original purpose, achieves either the same, or a different predictable pedagogical result in another educational context. Typical pedagogical requirements found in reusability definitions are independence, self-containment, or a defined learning objective.

More relevant for this thesis are **technical properties of reusability**. From the definitions of learning objects and reusability, three major requirements for successful reuse can be extracted: interoperability, availability and retrievability. As different definitions and meanings of these terms exist as well, a definition valid for this thesis for them is given.

Interoperability: Interoperability means that a reusable learning resource can be exchanged between two systems (e. g. authoring tools, repositories, learning management systems), which both comply with a common specification. [Hod02a, Pol03, DH03]

Availability: Availability of a reusable learning resource is given if it is published either publicly or to a restricted community, allowing users to physically access (i. e. use or download) the learning resource. [Wil00, Pol03, Kop03]

Retrievability: A learning resource is retrievable, if a user, who would benefit from this learning resource, is able to find it. Retrievability demands first adequate metadata for a learning resource, and second a retrieval system that makes use of this metadata for search interfaces. [Hod02a, DH03]

Existing approaches for reuse of learning resources interpret reuse as *simple reuse* – that is, applying a learning resource in a new educational scenario without changes to the learning resource. Changing a learning resource before reuse is not covered by existing definitions of reuse.

Furthermore, there is a gap between the hierarchical approach of content models and systems supporting reusability. All content models, which have been presented in the previous section, intentionally aim at enabling the creation of fine-grained contents, which can be reused in different combinations. This approach works fine within a given homogeneous system landscape. It is assumed that learning resources that are already aggregated into larger structures are still separately stored in a repository in order to be reused in another aggregation. A decomposition of aggregated structures into learning resources of lower levels of granularity is not intended.

The interoperability requirement leads to another concept: modularity. Can modularity be a criterion for reusability? And what means modularity with regard to learning resources? In order to answer these questions, a definition of modularity has to be found.

### 2.2.2 Definition of Modularity and Modular Operations

To approach what modularity means we take a look at the concepts of modularity in two other research areas: system design and software engineering. The first excursion takes us into the design of systems, such as computers or other complex industrial devices. According to Baldwin and Clark,

> "Two subsidiary ideas are subsumed in the general concept [of modularity]. The first is the idea of interdependence within and independence across modules. (...) The second idea is captured by three terms: abstraction, information hiding and interface."[BC99]

Modularity is seen as a design principle, which splits a complex system into several modular components, which each have less complexity. Modules have high internal and low external dependencies; they provide an interface for communication with other components of the system. Thus, the module concept of Baldwin and Clark is similar to the entities of content models in e-learning. The process of transforming a design by increasing its modularity is called modularization. Baldwin and Clark specify six basic modular operations, which are sufficient to describe all structural changes of an overall system. These six modular operators are [BC99]:

- Splitting a design into modules
- Substituting one module design for another
- Augmenting – adding a new module to the system
- Excluding a module from a system
- Inverting to create new design rules
- Porting a module to another system

With this set of modular operators, all structural changes of a system design can be described. If this notion of modularity is transferred to learning resources, a congruent set of structural operations should exist for the constitution of learning resources.

The second reference is modularity in software engineering. There, modularity is for example defined by Mikkonen and Taivalsaari as following:

> "Modularity is the property of computer programs that measures the extent to which programs have been composed out of separate parts called modules. A module is generally defined to be a self-contained component of a system, which has a well-defined interface to the other components. Something is modular if it includes or uses modules which can be interchanged as units without disassembly of the module. The internal design of a module may be complex, but this is not relevant; once the module exists, it can easily be connected to or disconnected from the system."[MT07]

Again, the existence of interfaces, self-containment, and interchangeability are important attributes. Furthermore, this definition demands that modules of a modular program can be exchanged without

disassembling the program. If a program is not yet modular, it can be modularized (according to Baldwin and Clark) in order to obtain a modular structure, which enables exchange (i. e. reuse) of modules.

Although reuse in industrial and software engineering is different from reuse in e-learning, the two presented concepts of modularity may indicate the direction for modularity considerations in reuse of learning resources. As in engineering, the reasons for modularization in e-learning are primarily the reduction of costs and complexity. Costs can be reduced by avoiding production costs for new learning resources as long, as the effort for reuse – retrieval costs, royalties, and re-purposing costs – is lower than the production costs for a new learning resource. Content models for learning resources already apply the concept of partitioning a learning resource for the same two purposes: reduction of complexity by standardized aggregation structures and reduction of costs by reuse. To support reuse, it is necessary to expose modular learning resources for exchange, to provide a clear interface, and to support abstraction in order to reduce complexity for creation and re-authoring. Abstraction of learning resources comes in the form of metadata. Ideally, it suffices to read the metadata record of a learning resource to know whether and how the learning resource may be used – though in practice metadata is mostly not precise enough.

What is missing in current content models and reuse definitions is the dynamic aspect of modularization that the concept of Baldwin and Clark contains. Baldwin and Clark incorporate that the breakdown of contents into modules can be changed by modular operators. Applied to learning resources this would mean that a learning resource can be repartitioned. It should be possible to transform parts of a learning resource into modules, to aggregate external modules, and to port modules within a learning resource to other learning resources. Such a modularity property as complement to reusability would give authors, tutors and other users more freedom of choice; they could chose the boundaries of modules to reuse. Derived from the discussed modularity definitions, modularity of learning resources is defined as follows:

**Definition 2.2 (modularity)**
*A content model or specification for learning resources is* ***modular****, if it allows to encapsulate, expose and separately reuse parts of a learning resource; these parts are called modules. Furthermore, modularity requires that modular operations can be performed on these modules.*

*By means of these modular operations, the content of modules and the structure by which modules are organized may change: the order of modules may change, modules may be removed, new modules may be inserted, fragments of modules may become modules themselves, and modules may be adapted to fit the surrounding modules.*

Which modular operations are required is still left to be defined. Based on the modular operators of Baldwin and Clark in conjunction with the open research issues of Duval and Hodgins [DH03] six modular operations for learning resources have been identified. These six operations allow a wide range of changes of learning resources. However, the list is not meant to be complete; further modular operations may be added. The six basic modular operations on learning resource are:

- Modularization
- Aggregation
- Exclusion (removal of modules)

- Replacement (substitute one module by another)
- Reorganization (changing the order of modules)
- Adaptation (transformation of a module)

Having defined modularity as a new property of learning resource formats, reusability could be redefined to put the envisioned choice of module boundaries into practice. The next subsection will analyze how the definition and requirements of reusability can be extended to make use of learning resource modularity.

### 2.2.3 Multi-Granularity Reuse

Modularity of a learning resource format has been defined in the previous section as the property of allowing to encapsulate, expose and separately reuse parts of a learning resource. This property is already more advanced than the notions of simple reuse that were examined before: the definition of modularity allows to partition and repartition a learning resource into reusable modules. However, the definition of modularity does not specify exactly which parts of a learning resource can be turned into modules. In this perspective, modularity is a weak definition with respect to the goal of supporting reuse of fine-granular learning resources. A stronger definition would claim that any part of a learning resource that is desired to be reused can be transformed into a module.

If the creation of a learning resource is based on an underlying content model (cf. 2.1.3), this content model specifies particular content element types of different levels of granularity that can be aggregated. It can be assumed that at least all these elements of a content model may be parts that someone might want to reuse. Ideally, reuse of learning resources of each aggregation level of an underlying content model should be enabled. In order to approach a suitable definition of reusability that regards multiple levels of granularity, we first define reusable fragments of learning resources:

**Definition 2.3 (reusable fragment)**
*A **reusable fragment** of a learning resource is defined as each part of a learning resource, which is potentially useful as either a separate learning resource or as a component for aggregation to a larger learning resource. If a learning resource has been created according to an underlying content model, at least all specified aggregation levels of the content model are considered relevant fragments.*

What a reusable fragment is depends on the underlying (original) content model. Generally, the definition is not restricted to a particular content model. It can be assumed that all aggregation levels, which are specified by a content model, are qualified for individual reuse. The granularity of a learning resource indicates its complexity, in both a technical and pedagogical sense. Granularity can mean either the aggregation level of a learning resource, or which educational functionality it provides. Multi-granularity refers to the fact that learning resources are aggregations of multiple other learning resources, which again may be aggregates, and so on. Content models (see Subsection 2.1.3) provide a formalization of aggregation levels and corresponding learning resource types. Hence, multi-granularity reusability requires that learning resources of all aggregation levels can be reused. This finally leads to the following formal definition of multi-granularity reusability:

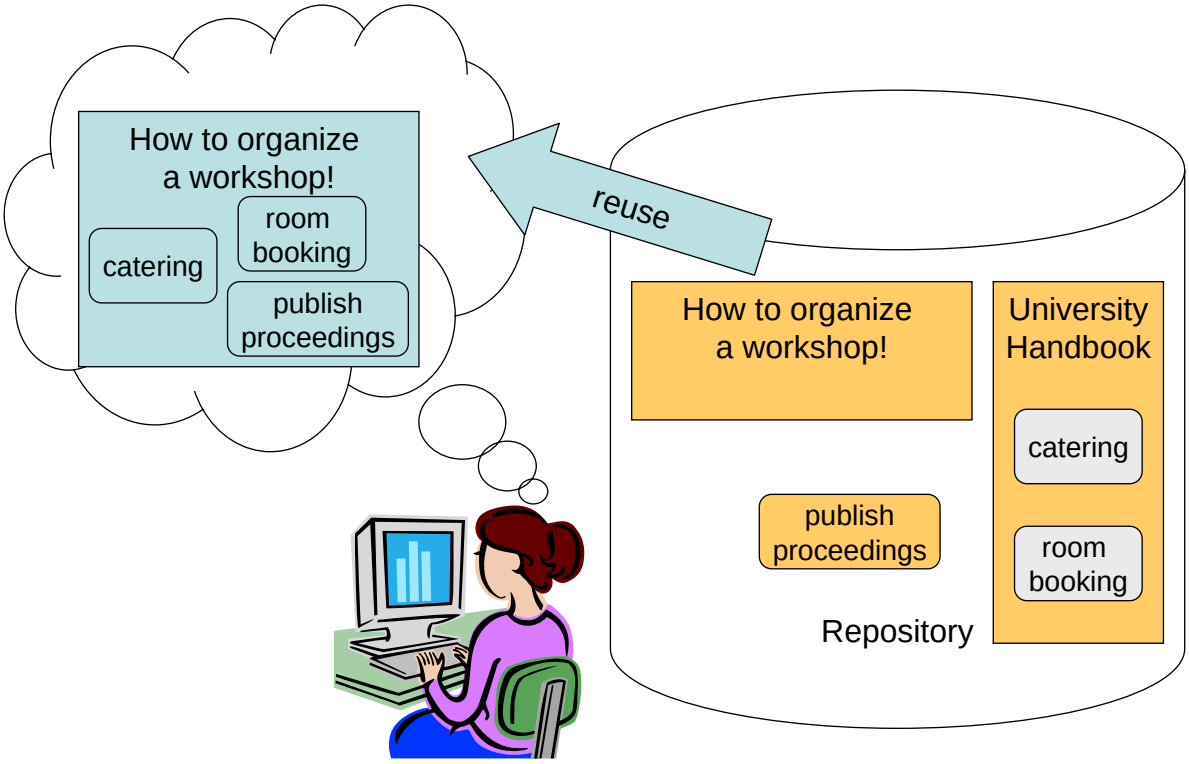


Figure 2.5: Use case for multi-granularity reuse.

**Definition 2.4 (multi-granularity reusability)**
***Multi-granularity reusability*** *of learning resources means to generally enable reuse of learning resources and all their reusable fragments at multiple levels of granularity as modules.*

The definition of multi-granularity reusability is stronger than modularity. It ensures that all learning resources of all aggregation levels of a content model can be reused. So far, we have a definition of multi-granularity reusability. There are some challenges for realizing this property in practice. The next section will describe these challenges and propose solutions.

## 2.3 Challenges and Own Approach for Supporting Multi-Granularity Reusability

In the previous section existing approaches of reusability have been analyzed. New definitions for modularity of learning resources and multi-granularity reusability were introduced. These definitions – when applied in practice – may arise some new challenges. This section analyzes which challenges are caused by the demand for modularity and multi-granularity reusability.

First an example and a scenario for multi-granularity reuse is given. Based on this scenario the challenges of multi-granularity reusability are analyzed. Afterwards, a new approach is proposed that serves as a foundation of actual solutions that are presented in subsequent chapters.

### 2.3.1 An Example for Multi-Granularity Reuse

Sarah works as a research assistant at a university. After having successfully organized a workshop about her research, she was asked to teach her colleagues how to organize a workshop. Sarah decides to create a Web based training course, which will be made available for her colleagues. Because her time is short, Sarah wants to reuse existing learning resources from other authors for her new course (see Figure 2.5).

In a public repository Sarah finds some learning resources about organizing events and particularly workshops. She finally decides to use a course that has been provided for free by a company. However, this course lacks some details that are important for workshop organization at Sarah's institution. For instance, the new course should contain descriptions of how to book conference rooms at the university, where to order a catering service, and how to publish workshop proceedings.

A guideline for publishing proceedings is offered by a publisher; it is suited to be reused by Sarah for her new course. The other two desired learning resources on room booking and catering are not so easy to find. Sarah searches for such learning resources but cannot find anything useful. Finally she asks a colleague who tells her that the university has a general learning resource called "handbook of university services" that describes the various services offered by the university administration. This handbook also contains information about booking conference rooms and ordering a catering service. Even though the handbook is available as a digital learning resource, Sarah was not able to find it in her first search because the details of the contents were not reflected in the metadata record of that learning resource.

Now that Sarah has obtained all required learning resources she starts to combine them into a new learning resource. First she uses the general learning resource on workshop organization as basis. The contents are mostly suitable, but Sarah has another structure in mind. She reorganizes the learning resource by changing the order of fragments; a few parts are removed because they are not relevant for her workshop scenario. Afterwards, she integrates the learning resource about publishing proceedings into the course. She also wants to aggregate the contents from the handbook of university services. In order to do so, she has to extract those fragments that she wants to reuse from the handbook. Finally, the contents of the resulting learning resource are done. But the learning resource now contains parts in different designs and different language styles. As a last step, Sarah adapts the learning resource to unify the appearance of the course.

### 2.3.2 A Scenario for Multi-Granularity Reuse in Heterogeneous Systems

This thesis is based on an understanding of reuse that involves heterogenous system landscapes. A special case of these heterogenous system landscapes is cross-organizational reuse, where multiple organizations, companies, or individuals exchange and reuse learning resources. In the example in Subsection 2.3.1 Sarah had to reuse learning resources that were created by other companies. It is assumed that not all actors involved in the exchange of learning resources use the same authoring tools, repositories or learning management systems.

Actors or organizations want to reuse existing learning resources for various reasons: it may save production costs, reduce maintenance efforts, or the required expertise for own authoring is not available. In all these cases, different kinds of reuse are thinkable. A user could directly use a learning resource

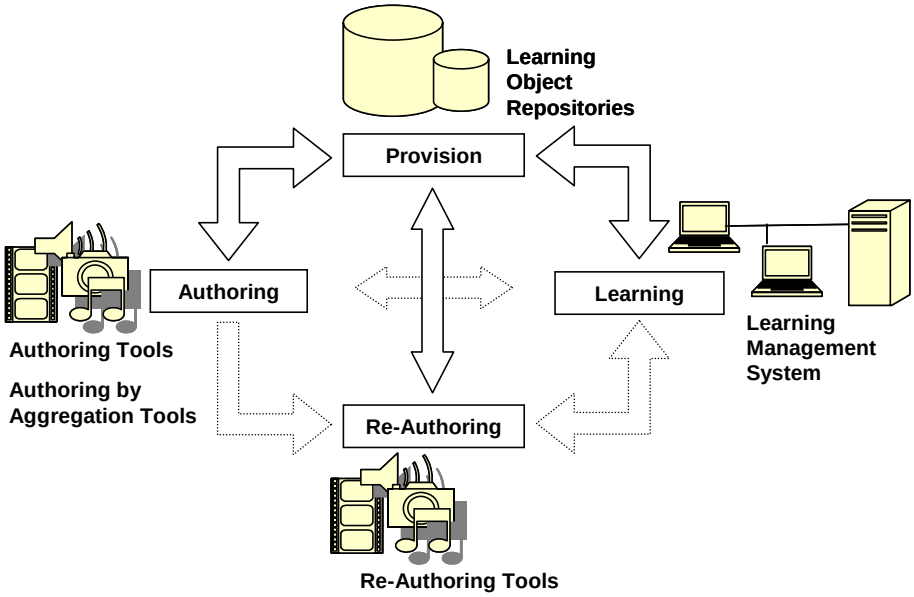


Figure 2.6: A learning resource lifecycle [RBH+05].

unchanged for teaching (this is the use case assumed by related approaches); but he could also make some modifications for adapting the learning resource for his particular purpose. A further use case is to aggregate either a whole learning resource or even only parts of it to a new learning resource. In order to differentiate between these different kinds of reuse in further chapters, they are defined here[11]. A more detailed description of and examples for different kinds of reuse can be found in [RZM+08].

**Definition 2.5 (reuse)**
***Reuse*** *of learning resources is every kind of use of existing learning resources, which are already used in a certain context. A learning resource may be reused in learning or teaching without any modification; a learning resource may be reused in authoring by aggregation; a learning resource may be reused in re-authoring.*

If a learning resource is not immediately applicable in a target educational setting, it has to be modified to better suit the requirements. A reuse that involves modification of the learning resource is generally called re-authoring. An extended learning resource life cycle, which explicitly includes a re-authoring phase, is illustrated in Figure 2.6.

**Definition 2.6 (re-authoring)**
***Re-authoring*** *is the modification of an existing learning resource before the learning resource is used in learning or teaching again. During the modification, parts of another (smaller) learning resource can be re-used. Re-authoring is independent of the reason for a modification of the learning resource.*

Modifications of a learning resource can be necessary for different reasons. An author might want to correct errors, which he recently found, before using a learning resource again. Similarly, some facts outdate over time and should be updated from time to time. Another motivation for modifications of

[11] The definitions of different kinds of reuse have originally been published in [RBH+05].

a learning resource is that it is intended to be used in a different educational context than before. The latter case is also called re-purposing.

**Definition 2.7 (re-purposing)**
***Re-purposing*** *is the transformation of a learning resource to suit a new learning or teaching context, which differs from the learning or teaching context the learning resource was created for.*

Re-purposing of a learning resource can be composed of three types of processes: modularization, aggregation and adaptation. These re-purposing processes are three of the six modular operations on modular learning resources. In addition, by means of aggregation, also creation of new contents may be involved in re-purposing.

**Definition 2.8 (modularization)**
***Modularization*** *means splitting a large learning resource into several smaller learning resources. The result can be either separate individual learning resources or an aggregation of reusable learning resources.*

**Definition 2.9 (aggregation)**
***Aggregation*** *is the process of combining small learning resources into a larger learning resource.*

**Definition 2.10 (adaptation)**
***Adaptation*** *means changing a learning resource with regard to one aspect to make it fit into a new context of use. Aspects are for example language, layout or terminology. To perform an adaptation an adaptation process is executed.*

The combination of all types of reuse – updates, corrections, modularization, aggregation, re-purposing for new-contexts – leads to a variety of versions of (almost) the same contents. Two types of versions can be distinguished: revisions (e. g. updates or corrections) replace previous versions, whereas variants are versions that are meant to exist in parallel (e. g. a Spanish variant of an English learning resource).

Reuse and re-authoring involve mainly four user roles: authors, teachers, learners, and content finishers. Authors originally create and distribute learning resources. Teachers obtain learning resources and use them for teaching. Optionally, learners may obtain a learning resource for learning, without involvement of a teacher. A content finisher is a special kind of author: he retrieves existing learning resources and performs a re-authoring in order to redistribute (i. e. resell) the modified contents.

While traditional multi-granularity approaches for learning resource reuse assume controlled authoring processes, tools and formats within a single organization, this thesis researches reuse and re-authoring also across different systems. That means, that tools, systems, formats, and pedagogical methods are no longer unified. Instead, a heterogeneous scenario is the normal case. Especially the following conditions are assumed:

- Different authoring tools
- Different pedagogical methods and content models
- Different learning object repositories
- Different learning management systems

Using different authoring tools also inevitably leads to the usage of diverse content formats and visual styles. Currently, this interoperability issue is solved by using a common exchange format. Most authoring tools are able to export a learning resource as a SCORM package, the de facto standard for Web based learning resources. Therefore, SCORM is also assumed to be the central exchange format that is understood by all involved systems of our scenario.

### 2.3.3 Definition of Requirements of Multi-Granularity Reusability

Multi-granularity reusability has been defined as enabling reuse of learning resources and all their reusable fragments at multiple levels of granularity as modules. In Section 2.2.1 three main technical requirements of reusability were identified for the case of simple reuse. In the scenario description, reuse was extended to cover different kinds of usage: simple reuse of an unmodified learning resource, but also re-authoring and aggregation. Thus, multi-granularity reuse has to become a granularity aware realization of simple reuse, as well as re-authoring, and aggregation.

Reusability is – as already discussed earlier – not only a property of a learning resource itself, but also of the system, which enables and supports retrieval and reuse. Proceeding from simple reuse to multi-granularity reuse, the impact of the system(s) on reusability increases; not only does the sheer exploding number of learning resources put higher demands on retrieval systems – also system support for modularization, aggregation and adaptation of learning resources is essential for successful reuse.

Basically, the technical reusability requirements that have been discussed in the previous section for reuse in general can be applied as well for multi-granularity reuse. Yet, they are not sufficient to completely ensure multi-granularity reuse. Some of the requirements have to be adapted and extended. For example, interoperability in the sense of multi-granularity reuse means more than being transferable and executable in a different learning management system – multi-granularity interoperability premises that an aggregation of multiple learning resources of similar complexity results again in a valid learning resource. As Section 2.2 has pointed out, there are three main properties that are often used to characterize the technical dimension of reusability. These properties are interoperability, availability and retrievability. Now, these properties can be concretized with regard to multi-granularity reuse.

**Definition 2.11 (availability)**
*All fragments of a learning resource are individually available for reuse, and especially for aggregation to larger learning resources.*

**Definition 2.12 (retrievability)**
*All fragments of a learning resource can be appropriately found and individually obtained by users. This particularly requires that adequate metadata for each fragment is either already available or can be automatically generated.*

**Definition 2.13 (interoperability)**
*All fragments of a learning resources can be separately reused, both individually and for aggregation. Aggregations of learning resources or fragments of learning resources are required to be sound learning resources themselves.*

These requirements are of technical nature; they do not directly address pedagogical issues. Assuming that reusability – whether simple or multi-granularity – always depends strongly on pedagogical skills of authors or teachers, this thesis leaves pedagogical tasks to human experts and focuses on technical issues.

### 2.3.4 Building Blocks for Multi-Granularity Reusability

Enabling multi-granularity reusability across different systems is the main goal of this thesis. The challenge is the combination of multi-granularity reuse of learning resources with the heterogeneity of systems, formats and styles. There are two main aspects that have to be taken care of: technical requirements for multi-granularity reusability and modular operations. Supporting multi-granularity reuse involves solutions that contribute to both aspects. The main technical aspects of reusability are *availability*, *retrievability* and *interoperability*. As modular operations especially the non-trivial operations *modularization*, *aggregation* and *adaptation* have to be supported. Figure 2.7 illustrates the fields to which this thesis provides contributions.

There are five building blocks which contribute to these fields. The building blocks are:

1. **Module concept.** A module concept is introduced in Chapter 3 that supports availability and interoperability of modular learning resources.

2. **Modularization.** In Chapter 3, existing approaches for modularization of learning resources are analyzed. Based on this analysis a generic reference model for modularization processes is specified. Modularization processes are considered to increase the availability of modular learning resources.

3. **Aggregation.** Aggregation as another modular operation is covered by Chapter 4. An existing aggregation process is extended. In addition, methods for improving retrievability in combination with aggregation are researched.

4. **Adaptation.** Adaptation is supported by an abstraction model for learning resource content representation and modification. This building block can also be regarded as a contribution to interoperability, because it is a basis for unifying aggregated learning resources towards a consistent appearance.

5. **Metadata generation.** Retrievability of learning resources is improved by the development of a new automatic metadata generation method in Chapter 6. This method allows the classification of learning resources according to their topics.

(1) The SCORM specification and current implementations of LORs do not support reuse and retrieval of fragments of learning resources sufficiently. Although multiple levels of granularity can be represented in SCORM, no support for separately reusing these fragments is provided; only whole packages can be retrieved from learning object repositories. This thesis researches how reusability of fragments can be improved and presents a module concept as an extension to SCORM in Chapter 3. The module concept allows to explicitly expose fragments of a SCORM package for separate reuse.

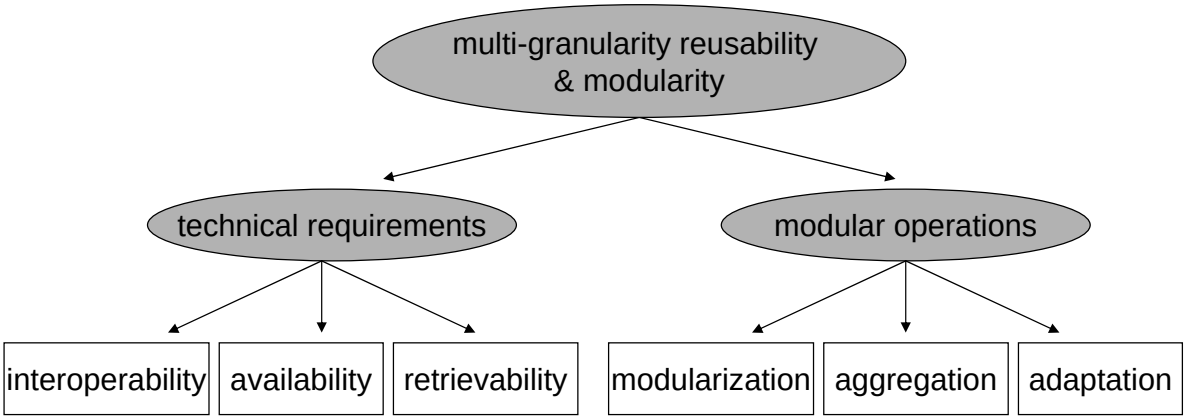


Figure 2.7: Requirements of multi-granularity reusability.

(2) Once such an extended standard is available, existing learning resources are still available only as monolithic SCORM packages. Therefore, Chapter 3 also discusses how existing learning resources can be transformed into aggregates of modular, reusable learning resources.

(3) When fine-grained learning resources are finally available, aggregation of these learning resources gets into the focus. Authoring by aggregation is an authoring paradigm foremost based on creating a learning resource out of existing ones of finer granularity. Chapter 4 shows how authoring by aggregation can be applied to heterogeneous systems. Furthermore, retrieval as part of this authoring paradigm is better integrated into the authoring process and improved by usage of context information about the learning resource being created.

(4) A learning resource that is aggregated of several learning resources from different sources is likely to appear as a patchwork: design, terminology, didactic style, etc. probably show differences, which may interfere with the usability of the resulting learning resource. Thus, aggregated learning resources have to be adapted to fit together. As a foundation of adaptations of learning resources, the architecture of a re-purposing framework is specified in Chapter 4, upon which re-purposing applications can and have been efficiently developed.

(5) Last but not least, retrievability strongly depends on good metadata. As authors often do not provide enough metadata, additional metadata has to be generated by automatic methods. A new method of automatic metadata generation is developed and evaluated in Chapter 6: The knowledge of many people, collected in the online encyclopedia Wikipedia, is used as a basis for determining the topic of a learning resource.

## 2.4 Summary

In the beginning of this chapter, related work in the area of e-learning has been presented and discussed. Several common definitions of learning resources have been compared. Along with learning resources, reusability, learning object repositories and content models were explained.

Analyzing the scopes and limits of current related work on reuse of learning objects, a gap has been identified between the concepts of multi-granularity reusability and modularity in closed systems and the interoperability issue in heterogeneous systems. Multi-granularity reusability is currently not suffi-

ciently supported for heterogeneous systems. Multi-granularity reuse is generally addressed by content models, which specify multiple aggregation levels, allowing to aggregate learning resources of one level of granularity into larger units. However, whenever a learning resource leaves the boundaries of a homogenous system, it is transformed into formats, which may no longer support efficient reuse of multiple granularity levels at the same time.

Thus, improving support for multi-granularity reusability has been identified as the main goal of this thesis. Multi-granularity reusability can be supported by improving technical reusability requirements (availability, reusability and interoperability) on the one hand and modular operations on learning resources (modularization, aggregation and adaptation) on the other hand. Five building blocks that each support one or more of these aspects have been defined. These five building blocks are researched in this thesis to finally achieve more efficient support for multi-granularity reuse.

# Part II

# Supporting Modular Operations: Modularization, Aggregation and Adaptation

# 3 A SCORM Module Concept for Supporting Modularity and Modularization

Two technical requirements for reuse of learning resources are availability and interoperability (see Figure 3.1). In the case of simple reuse, the challenges of availability and interoperability are solved: learning resources are stored and published in a learning object repository, from which they can be retrieved and aggregated [Hoe05, BLW99]. Ensuring availability and interoperability in the scenario of this thesis – multi-granularity reusability in heterogeneous systems – is more complex.

Interoperability between heterogeneous systems is commonly solved by relying on SCORM as a de facto standardized exchange format. However, exporting a learning resource from an original format into a SCORM package foils the modularity property of all those learning resources the package consists of. These fragments originally were modules that could be separately retrieved from a repository. After the export they are no more modules; the fragments can be reused only as part of the overall SCORM package.

The first contribution of this chapter is a module concept that extends SCORM to support modularity that is introduced in Section 3.1.2. The module concept enables SCORM packages to consist of several modules that can be reused separately. A second contribution is the definition of a process model for modularization of learning resources in Section 3.2. Modularization means foremost to split a complex learning resource into several reusable components. After this processing, the components are separately reusable. Several existing modularization approaches are analyzed and a generic process model is derived.

## 3.1 A Module Concept for SCORM-Compliant Learning Resources

There are already modular learning resource specifications. Could such a specification be used? Or could a new specification be developed from scratch that ideally realizes all those requirements that were defined in the previous chapter? In theory such a specification could perfectly match the modularity

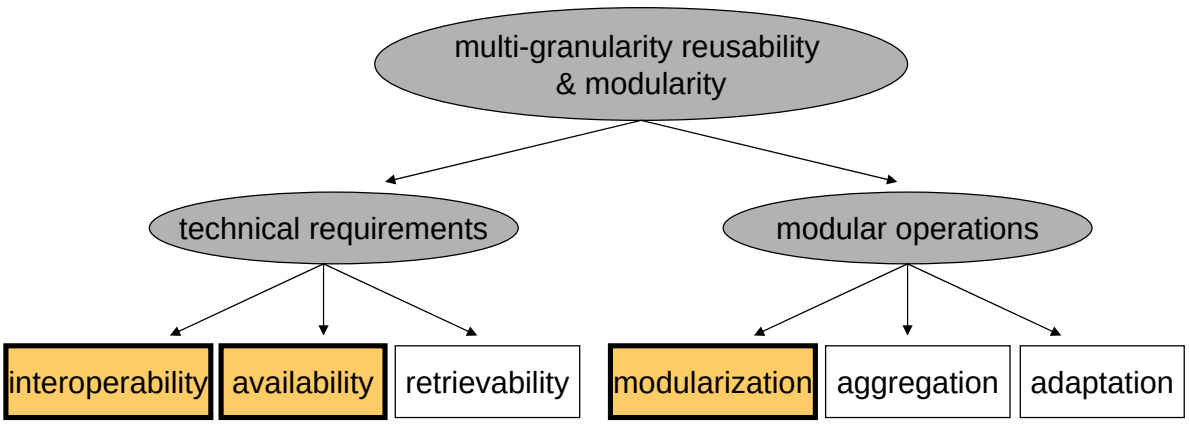


Figure 3.1: Contributions of Chapter 3 to multi-granularity reusability.

requirements. But the heterogeneity conditions of the underlying scenario would not be met. A heterogeneous system scenario, such as Chapter 2 describes, requires that a module concept considers the restrictions of all involved components. A module concept that would break compatibility with existing authoring tools, learning management systems and learning object repositories would be a badly designed concept. There are basically two ways to ensure compatibility with existing systems: either by developing format transformations for all involved learning resource formats, or by relying on SCORM as a de facto standard that is already supported by most relevant systems. In practice, SCORM is the de facto standard for Web based learning resources. Almost each system for Web based e-learning – whether authoring tool, repository or learning management system – is able to handle SCORM compliant contents. This wide-spread SCORM compliance is a welcome situation, which may support the interoperability of modular learning resources. Breaking with SCORM compliance would nullify this opportunity.

Thus, SCORM has been chosen as the base specification for modular learning resources. First, the SCORM specification is analyzed regarding its current support for modularity. Afterwards a concept for modular learning resources as an extension to SCORM is developed. Finally, the module concept is reviewed regarding its suitability for modular reuse.

### 3.1.1 Analysis of Modularity in SCORM

SCORM has been chosen as basis for a module concept. Therefore, the SCORM specification has to be compared to the requirements of modularity. Modularity of a learning resource specification is defined by two properties: first it has to enable encapsulation, exposure and separate reuse of parts of a learning resource. And second, modular operations have to be possible. SCORM would be considered a modular specification if it fulfills these two properties.

First, the requirement of encapsulation, exposition and separate reuse of fragments is to be checked. A whole SCORM package is encapsulated and exposed via an interface. SCORM as well defines a content aggregation model and provides interface specifications for fragments of a SCORM package (SCOs, assets, activities). Thus, these fragments can be encapsulated and exposed. However, the SCORM specification does not meet the requirement of enabling the separate reuse of these fragments. Therefore, SCOs, assets and activities of a SCORM package are not modules according to the above definition. Even though a SCORM package has an inner structure that reflects structural aspects of an original content model, the aggregation elements of a SCORM package are not full-fledged modules (these elements lack the property of separate reusability).

Content aggregation elements of SCORM are SCOs, assets and activities (so called *items* in a manifest). All of these element types are specified in the manifest of a SCORM package and may optionally be described by a separate metadata record. However, such an element cannot be extracted for separate reuse or exchanged without a thorough analysis of the actual contents. The reason for this nuisance is that the actual content files are sometimes not assigned to the corresponding aggregation element. Even worse, some files, such as style sheets and particular images, are used by multiple elements. It is not necessarily bad to use only a single instance of a style sheet or logo for a whole learning resource; on the contrary, it often makes sense to avoid redundancy. But if parts of a SCORM package should be handled

as modules (e. g. a particular SCO is worth to be separately reused, replaced or updated), these shared resources impair modularity.

### 3.1.2 Requirements on a Module Concept

As the analysis in the previous subsection has shown, SCORM does not fully meet the modularity property. What SCORM particularly lacks is the exposition and separate reusability of fragments. Although SCORM uses a content model, the elements within a SCORM package are not ready to be separately reused. Aggregation elements of a SCORM package can neither be individually retrieved from learning object repositories, nor can they be immediately reused without the containing package. Other content models discussed in Chapter 2 meet the modularity requirements, but cannot be used in a heterogeneous scenario, because they are understood only by few authoring tools and learning management systems.

Therefore, a new concept of modular learning resources is needed, which suits both, the theoretical demands of modularity and the practical situation of a variety of existing systems for creation, distribution and usage of learning resources. The module concept has to particularly establish a separate reusability of fragments of a learning resource, as this is a shortcoming of the current SCORM specification. But other properties have to be fulfilled by a module concept as well. One of these properties is, for instance, the downward compatibility of a new module concept with standard SCORM implementations. Also a separate exposition of learning resource fragments via individual metadata records has to be supported. In total, six requirements have been identified that have to be fulfilled in order realize modularity in SCORM. These requirements are

1. **SCORM compliance.** The first requirement – SCORM compliance – means that a system, which does not support the modularity extension, can handle a modular learning resource as a standard SCORM package. This feature ensures interoperability with existing systems.

2. **Modularity requirements (according to Section 2.2.2).** Modular learning resource have to meet the modularity requirements that have been specified above. The concept has to enable that a learning resource may consist of several modules of different complexities. Ideally, each aggregation element of the SCORM CAM could be a reusable module.

3. **Support for metadata.** It is undoubted that learning resources have to be described with metadata for enabling efficient reuse. Consistently, it is necessary that each module within a modular learning resource is described by and delivered with a separate metadata record.

4. **Enable modularity-awareness of repositories.** And finally, the concept has to support that repositories can be aware of modularity and modules. If a learning resources stored in a LOR is aggregated of multiple modules, the repository should know about and make the modules separately accessible. Users should be able to find and download these modules independently of the enclosing learning resource.

5. **Support for modular operations.** The concept must be also designed to allow modular operations on learning resources. These operations have been named in Section 2.2.2. Modular operations are, for instance, modularization, aggregation or adaptation.

6. **Support for versions and updates.** The module format should also support the existence of multiple versions of the same module. Versions can be divided into revisions, which are consecutive updates replacing the previous revision, and variants, which are parallel versions (e. g. a German variant of an English learning resource). Different versions of one module may exist at the same time. When a new version of a module becomes available users should have the choice to optionally substitute the old with the new version. When a module is changed, a new version – either a revision or variant – should be automatically generated.

These requirements may serve as a guidance for the specification of a module concept that combines modularity of learning resources and a downward compatibility with existing SCORM compliant systems.

### 3.1.3 The Module Concept

The concept for modular learning resources extends SCORM in order to fulfill the requirements specified above. For the rest of this thesis the use of SCORM version 1.2 is assumed; this is still the most commonly used version, although a newer SCORM specification (SCORM 2004) has been existing for some years. The resulting concept has three parts: first, it consists of a rather conceptual understanding of what modular learning resources should be like; this part is reflected mainly by the requirements and by some further properties that can be observed from the actual implementation. Second, the concept comprises a tangible realization in the form of a specification of an extension to the SCORM format. And third, the concept includes also instructions for how modular operations on these learning resources can be performed.

#### Combining the Concept of Modules with SCORM

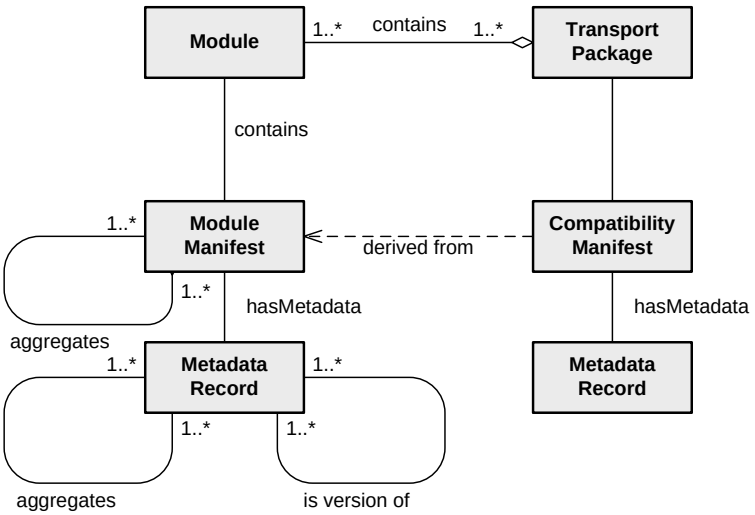


Figure 3.2: UML diagram of module concept.

There are two possible approaches for aggregation of learning resources: aggregation by copy and aggregation by reference. Aggregation by copy is what SCORM does right now – all contents are copied

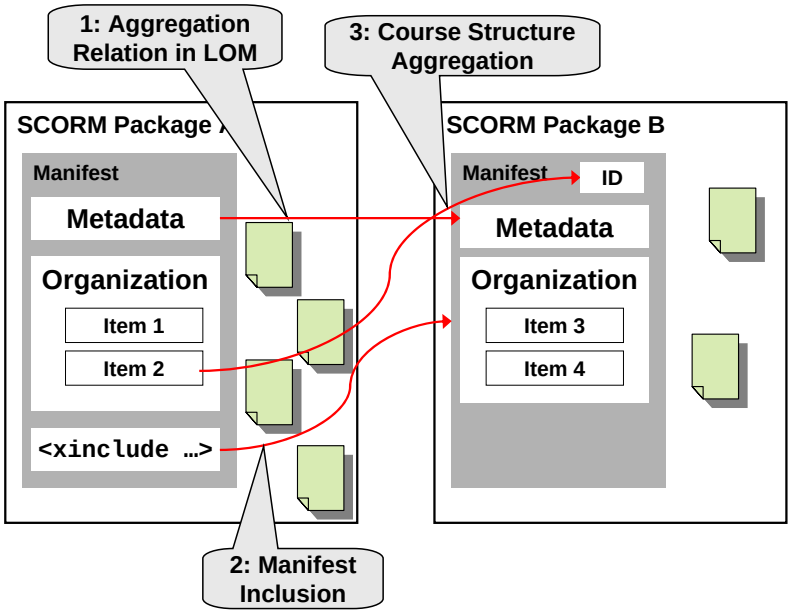


Figure 3.3: Three levels of aggregation.

into a SCORM package and cannot be easily separated afterwards. Other content models, for instance the ResourceCenter content model, represent the aggregation of learning resources as references [Hoe05]. The concept proposed in this thesis is based on aggregation by reference; modules are aggregated by referencing the included module (see Figure 3.2). The involved systems have to ensure that all required modules are correctly delivered in the end. The advantage is that modules can more easily be aggregated, updated, or separately reused. In order to enable references and the tracking of different versions of modules, each module needs to have a unique identifier.

The basic idea is that all modules virtually share a common file space. Thus, relative references between modules are possible; for instance, an HTML page of one module may link to an image located in another module (given that both modules are delivered together). Adapted from the learning resource format of the L3 project [TA03], each module is supposed to have a distinct namespace. Two kinds of namespaces are envisioned: the first one is derived from the internet domain system: similar to the naming of, for instance, packages in the Java programming language, folders are named by domains and sub domains belonging to the author. An author working for SAP could for example create a module in the folder "/com/sap/module42". Because domain names are used, it is unlikely that namespace conflicts occur. The second namespace convention is a flat model, where the module identifier is used as the folder name. The advantage is that module identifiers have to be globally unique, anyway. On the other hand, these folder names can hardly be associated with their probable contents by humans. The module concept allows both namespace concepts in parallel. However, the identifier based namespace is preferred, because it avoids the risk of duplicate folder names.

For the implementation a module is equivalent to an unzipped SCORM package moved into a particular sub folder relative to a root (this sub folder is the module namespace). Thus, the obligatory SCORM manifest is also located in that folder. Because a SCORM manifest may contain a LOM record, this is the place to put the module metadata. Aggregation of modules is realized by references between manifests and entities within them; this is described in more detail below. The relation category of LOM

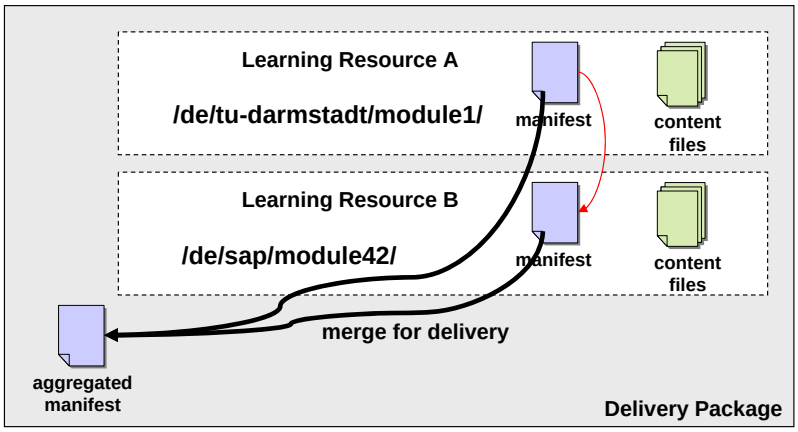


Figure 3.4: Bundling a SCORM-compliant package.

is intended to express references to other learning resources [Hod02b]; the SCORM specification is also designed to specify references between content elements. The novel approach of this module concept is the combination of three different types of references and the semantic assigned to these references. For distribution of modules, the contents of one or multiple modules are packaged into a zip archive file. Generally, it is possible to package any modules together, no matter whether they are related to each other. In practice, an archive will normally contain only modules that belong together. Particularly, when a module is needed to be transported to another repository or tool, all dependencies to required modules are recursively resolved and included in the transport package. The separation of folder namespaces ensures that no conflicts of overlapping modules occur. A transport package may also be transformed into a compliant SCORM package to support modularity-unaware systems. In this case, a flat manifest file is generated, which describes the structure of a whole module and included modules in a single manifest document.

The aggregation by reference of modules is performed on three layers (see Figure 3.3). These three layers serve different purposes. The first layer is the metadata layer: The LOM record of a module contains typed relation entries for all directly included modules. The relation entry contains only the identifier of an included module and can be used by repositories and other systems to easily identify which modules are to be delivered together as a bundle. The second layer, manifest aggregation, uses an option of the IMS Content Packaging (IMS CP) specification [IMS01]. IMS CP optionally allows to use so called *xinclude* tags [MO06]; xinclude is a mechanism for merging multiple XML documents into a single one. For the aggregation of modules, *xinclude* tags are used to merge all manifest documents of included modules into the aggregating module manifest. In fact, a virtual manifest for the whole aggregated learning resource is generated[12]. Within this virtual manifest, it is now possible to utilize the activities, SCOs and resources of other modules inside the organizational structure of the aggregating module. This utilization finally is the third aggregation layer. By integrating activities or SCOs of other modules into the course structure, aggregation of learning resources can be achieved.

[12] The virtual manifest is a document that is present only in main memory while tools process aggregated modules. However, the virtual manifest may become a physical document (the compatibility manifest) when an aggregated learning resource is exported as a SCORM compatible exchange package.

### Compliance with the SCORM Specification as Fallback

An important requirement is SCORM compliance in order to be interoperable with existing systems. Therefore, the modularity extension supports compatibility manifests. It has already been mentioned above, that virtual manifest documents are constructed by means of *xinclude* tags. For compatibility reasons, it is also possible to process the inclusions and generate a single physical manifest file for the whole aggregated learning resource. This compatibility manifest can be stored in the root of a a distribution package as a regular SCORM manifest. While an compatibility manifest is ignored by modularity-aware systems, the package is at the same time considered a conform SCORM package by modularity-unaware systems. Thus, a modular transport package can be processed by both, aware and unaware systems. The creation and location of compatibility manifests is illustrated in Figure 3.4.

A compatibility manifest is marked as such in the metadata record. Therefore, modularity-aware repositories can detect a modular transport package and treat it in a special way. The transport package is split into its constitutive modules and the compatibility manifest gets discarded. As module folder namespaces are assumed to not be nested the repository has to recurse into all folders of the package until a manifest is found. Each folder containing a module manifest is treated as a distinct module including all subfolders. Thus, repositories are enabled to manage, provide and make retrievable all modules separately, once they are uploaded to the repository.

The creation and tracking of versions is also realized via metadata relations. Whenever a revision or variant is derived from an existing module, the module identifier of the original version is written as a predecessor relation to the metadata record of the new version. As long as all consecutive versions of a module are available in one repository, it is possible to track the evolution of versions of a module. Hence, if a user wants to see whether his formerly obtained modules are outdated, he may query a repository for newer revisions.

### Realization of Modular Operations

In Section 2.2.2 six modular operations on learning resources have been identified. These operations should be – according to the requirements – be supported by the module concept proposed in this section. Therefore, a description of whether and how these operations are supported is given below. The six modular operations are schematically illustrated in Figure 3.5.

- **Modularization.** Modularization is the process of splitting a learning resource into multiple smaller modules. As SCORM is the base format, especially a separation of existing aggregation elements (SCOs, assets, activities) is of interest. For a successful modularization of these elements, the dependencies between activities, SCOs, assets and finally resources have to be resolved. All resources an element depends on have to be identified and transferred to an outsourced module. Often, some files are required by several elements – in that case, the resources have to be copied to all modules that require these resources. The analysis of dependencies cannot rely on the SCORM manifest alone. We have observed that sometimes not all required physical files are correctly assigned in the manifest document. Therefore, it is necessary to parse content files (e. g. HTML documents) for references and inclusions of other files. In some cases, other module boundaries than the internal

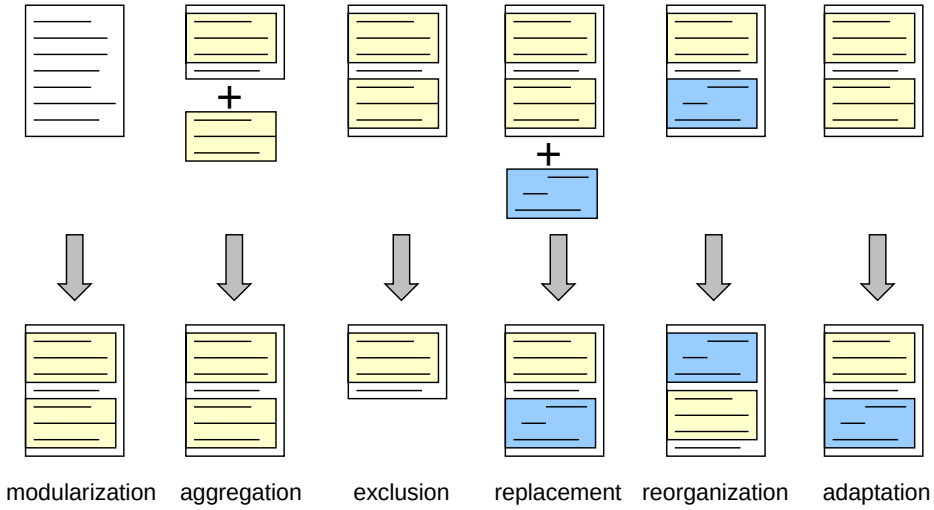


Figure 3.5: Schematic illustration of modular operations.

structure of a SCORM course are desired. A SCO often consists of multiple HTML documents. If these documents are desired to be separatly reused, modularization becomes more complex. An extensive analysis of content files and their internal structure (e. g. links and menus of HTML conglomerates) is inevitable. Modularization can then be implemented in two consecutive steps. In a first step, the internal structure of a SCO is externalized into the SCORM manifest; the second step transforms the elements into separate modules.

- **Aggregation.** The aggregation of modules is well supported on the level of SCORM elements. The aggregation of manifests allows to reuse contents from single resources over assets and SCOs to whole activities in the context of a larger learning resources. These elements can be used equivalently to local elements by modularity-aware authoring and re-purposing tools.

- **Exclusion (removal of modules).** Exclusion of modules is a trivial operation. A module is excluded by removing the links on all three aggregation levels.

- **Replacement (substitute one module by another).** Replacement of modules is mainly required for either applying an update (revision) for an already used module, or for substituting a module by a more suitable variant. But a replacement with a completely different module is also possible. Technically, a replacement is performed by changing the aggregation links. On two of the layers – the relation metadata and the manifest inclusion – a replacement is equivalent to the removal of the old module and the aggregation of a new one. Only the third layer is more challenging. Elements of the two modules (old and new) might have different identifiers. Another challenge is that references between content documents have to be rewritten: for example a HTML document from one module includes an image or animation from a different module. When the module containing the image is replaced by a new version, the reference to the image has to be changed in the HTML document. These kinds of operations make an analysis and modification of content documents necessary.

- **Reorganization (changing the order of modules).** Reorganization of modules is primarily a structural modification of the SCORM manifest of an aggregating module. In fact, the aggregation of

modules is not ordered as such. Only the third aggregation level, element references within a manifest, affects the order in which contents are presented to the learner. In this regard, a reorganization of modules is practically a simple reorganization of aggregation elements in a SCORM manifest.

- **Adaptation (transformation of a module).** The last modular operation to be supported, adaptation of a module, means that a module is replaced by a newly created version derived from that module. Thus, adaptation breaks down into creating a new variant of an existing module and replacing the module's original version by the new one in an aggregate. Technically, the module is simply modified and the metadata records of the adapted module and the aggregating module are updated.

As the description above has shown, the presented module concept enables the six relevant modular operations to be performed. On the level of modules, the basic operations *splitting*, *aggregating*, and replacement are the foundation of all modular operations. In addition, some processing of SCORM manifests and content documents is required to complete the realization of all six modular operations.

### 3.1.4 Review of Requirements for a New SCORM Module Concept

The intention behind the module concept is to enable modular reuse of learning resources in heterogeneous systems. Is this goal achieved by the developed module concept? That question is analyzed by checking if the previously defined requirements on such a concept are matched.

1. **SCORM compliance.** SCORM compliance ensures continuity by interoperability with existing systems. The module concept supports the creation of a compatibility manifest for being SCORM compliant. SCORM-enabled systems can handle a modular transport package as a traditional SCORM package.

2. **Modularity requirements.** Modularity is extensively supported by the concept. It allows the creation of modular learning resources, which are composed of several modules of varying complexities. These modules cannot only be used as part of the original aggregation, but can be made available separately by learning object repositories. It is then possible to use these modules as stand-alone contents or to integrate them into another learning resource.

3. **Support for metadata.** Each module contains a metadata record according to the LOM standard. The metadata record of the top module of an aggregation hierarchy is at the same time understood as the effective metadata record of the whole aggregate. Thus, metadata is supported for both, the modularity mode and the SCORM-compliance mode.

4. **Enabling modularity-awareness of repositories.** When a modular transport package is uploaded to a repository, it can be identified by the repository, because the compatibility manifest is marked. The individual modules can be extracted from the transport package and handled by the repository as separate objects. For delivery of a module, all aggregation dependencies from the metadata records have to be resolved recursively to ship all required modules in one package. Also, the repository has to create a new compatibility manifest during the delivery process.

5. **Support for modular operations.** Modularity also comprises support for modular operations; only if modules can be processed by an effective set of operators, modularity is actually ensured. All six relevant modular operations on learning resources are supported by the concept. For each of the operations the previous subsection has proposed a feasible implementation of that operation on the basis of the modular concept.

6. **Support for versions and updates.** Revision control is supported by new types of metadata relations. A version is technically a new module, which knows its immediate predecessor. By collecting the version relations of all modules within a repository, the history of a module can be tracked and presented to the user. A drawback of storing only the immediate predecessor in the metadata record is that version chains may get broken. If not all modules in a version chain are available in the repository, it becomes hard to find newer versions of a given module. Updates are supported by tracking revisions of a module and afterwards performing a replacement operation of the module.

All six requirements are fulfilled by the presented module concept. There is a minor shortcoming of the versioning system: the whole sequence of versions has to be available in order to track these relations from end to end. An alternative could be to store the whole history (all predecessors) in every module; however, this could lead to much larger metadata records.

Chapter 7 describes an implementation of the module concept and a tool that supports modular operations on learning resources. In that chapter, the module concept is evaluated regarding its usefulness in practice.

## 3.2 Modularization Methods and Processes

An important modular operation is the modularization of learning resources. Modularization is the process of splitting a learning resource into an aggregation of multiple modules. This section deals with this modularization operation, which is also called post-production modularization to distinguish it from a systematic design and authoring of modular learning resources. First, post-production modularization is discussed in general. Afterwards, a reference process model for modularization is introduced. Finally, also a concept for automatic modularization is presented.

### 3.2.1 Existing Modularization Methods

Modularization is the process of transforming a learning resource into an aggregation of multiple smaller learning resources. The necessity for modularization of learning resources is highlighted for example by Duval and Hodgins [DH03], though they name this process *decomposition*. The difference between modularization and decomposition is that modularization results in the same learning resource with a higher degree of modularity, whereas decomposition produces multiple unconnected learning resources.

Some authoring tools for learning resources also support splitting SCORM-based learning resources into smaller ones. An example for such a functionality is the Reload Editor[13]. But this functionality is restricted to selecting a subtree of the SCORM package and manually exporting it as a new package.

[13] http://www.reload.ac.uk/editor.html

The result is not a modular learning resource (in such a way that it is a learning resource consisting of multiple smaller learning resources), but only an exported part of the original course. Another authoring tool is the Phoenix editor [FMMF05], which is used to create documents enriched with pedagogical markup. Phoenix supports the decomposition of a document into multiple pedagogical units, if they have been marked as such with markup. Again, no modular learning resource is produced as a result of the decomposition process. Another example for decomposition is the Abstract Learning Object Content Model (ALOCoM) framework [VJD+06]. ALOCoM comprises a content model, a storage infrastructure for learning resources conforming to that content model, and decomposition tools. The core idea of ALOCoM is that a learning resource is decomposed into small fragments. These fragments are stored in a central repository, and can be reused in the authoring process of new learning resources. The main tools for decomposition and aggregation are plug-ins for Microsoft PowerPoint and Word, but other formats, such as SCORM, are also supported.

Three more approaches have to be mentioned, that aim at a slightly different goal. They transform arbitrary resources into a more pedagogical and formalized representation. Doan et al. extract learning resources from Web pages by applying pattern matching algorithms. They transform the contents and store the resulting learning resources in a repository [DBD06]. A similar approach is the workflow embedded authoring that Rostanin describes [RL07]. He identifies that especially small companies need to integrate learning into work processes. But not only learning, also authoring is integrated by Rostanin into workflows. He assumes that learning objects need not be created from scratch, but can be extracted from existing documents or Web pages. These documents are transformed into fine-grained learning objects, which are suited for workflow embedded learning. The third related work on transformation of learning objects is a guideline by Doorten et al. [DGJ+04]. They describe how learning contents from traditional university education can be decomposed into reusable learning objects. This guideline focuses on how the pedagogical usefulness of the resulting learning resources can be achieved. The decomposition is reported as a manual process that has to be performed by a subject matter expert.

### Segmentation of Multimedia Content

Aside from the e-learning field, there are some further methods worth to be mentioned: segmentation methods. Segmentation means splitting a resource into several segments. Segmentation methods are known particularly for video, audio and text resources.

Digital video segmentation is "the problem of decomposing a video into its component units" [HWJ94]. Typically, a video is decomposed into shots; segmentation methods aim at identifying the boundaries between subsequent shots. The identified segment boundaries can for instance be used for creating better visual presentations of a video [Man98, Ste99].

Audio segmentation is performed slightly different. In contrast to video segmentation, boundaries between multiple recording phases are seldom useful. Audio segmentation can be based on different concepts, for instance different speakers, topic of speech, or audio type (music, speech, silence, etc.). Segmentation by speaker or audio type are typical classification tasks, which can be performed by machine learning methods (e. g. support vector machines) using audio content based features [MN03, LLZ01].

Segmentation by topic first requires a transformation; an audio stream is transcribed into a textual representation of the speech. Afterwards, the segmentation algorithm is applied to the transcript [PC97].

Thus, segmentation of audio streams based on transcripts leads to another type of segmentation methods: text segmentation. Text segmentation methods principally split texts into multiple coherent blocks. While some methods utilize known linguistic structures, such as sentences or paragraphs, others regard a text simply as a sequence of words. The task of a text segmentation method is to identify boundaries between blocks of text, where the text within each block is coherent, and two adjacent blocks are relatively incoherent. Two exemplary text segmentation methods are the one by Ponte and Croft [PC97], and the TextTiling algorithm by Hearst [Hea94]. Both methods perform the segmentation in three steps. First the text is divided into a large number of small text blocks. Then, for each pair of adjacent blocks a similarity score is calculated. The resulting distribution of similarity scores over the whole text is used to identify those positions, where coherence is lowest; these positions are interpreted as segment boundaries.

Documents consist not only of a sequence of words, but also of structure. This structure can be either markup or a visual layout. Document can be segmented either based on an analysis and available knowledge of the structure, or by an analysis of the visual arrangement of texts and other elements. Layout analysis relies on the discovery of geometric and linguistic patterns in a visual representation of a document [Sum98]. Often, optical character recognition is performed before as preparation. Analysis of markup can be used for instance for separating real contents of a document (e. g. a Web page) from advertisements, menus and other irrelevant parts [KY07]. Markup-based segmentation methods can be divided into site-independent methods and methods that consider only Web pages of the same site.

The difference between segmentation methods and modularization is mainly the structure of the respective documents. Multimedia contents, such as video or audio documents are linearly organized, whereas learning resources typically provide a tree structure. In audio and video documents, there are breaks, such as shot boundaries in video footage or topic shifts in audio streams. By contrast, learning resources often cover a single topic; module boundaries are aligned with pedagogic structures.

### 3.2.2 Post-Production Modularization

The focus of this section is post-production modularization, which is one of the previously described modular operations on learning resources. Thus, post-production is concerned with the splitting of already existing learning resources into multiple modules. Some people use the term *modularization* for the systematic design and creation of modules from scratch. This meaning is not considered within this thesis. Modularization is regarded strictly as the process of splitting an existing learning resource into several modules.

In our example from Subsection 2.3.1 there is a *handbook of university services* that contains some parts, which are desired to be reused by Sarah. This example highlights our notion of modularization: Sarah has to modularize the handbook. As a result of the modularization process the contained parts on *booking conference rooms* and *ordering a catering service on the campus* are made available as separately reusable modules.

There exist different approaches for post-production modularization. They can be classified according to different aspects, for instance

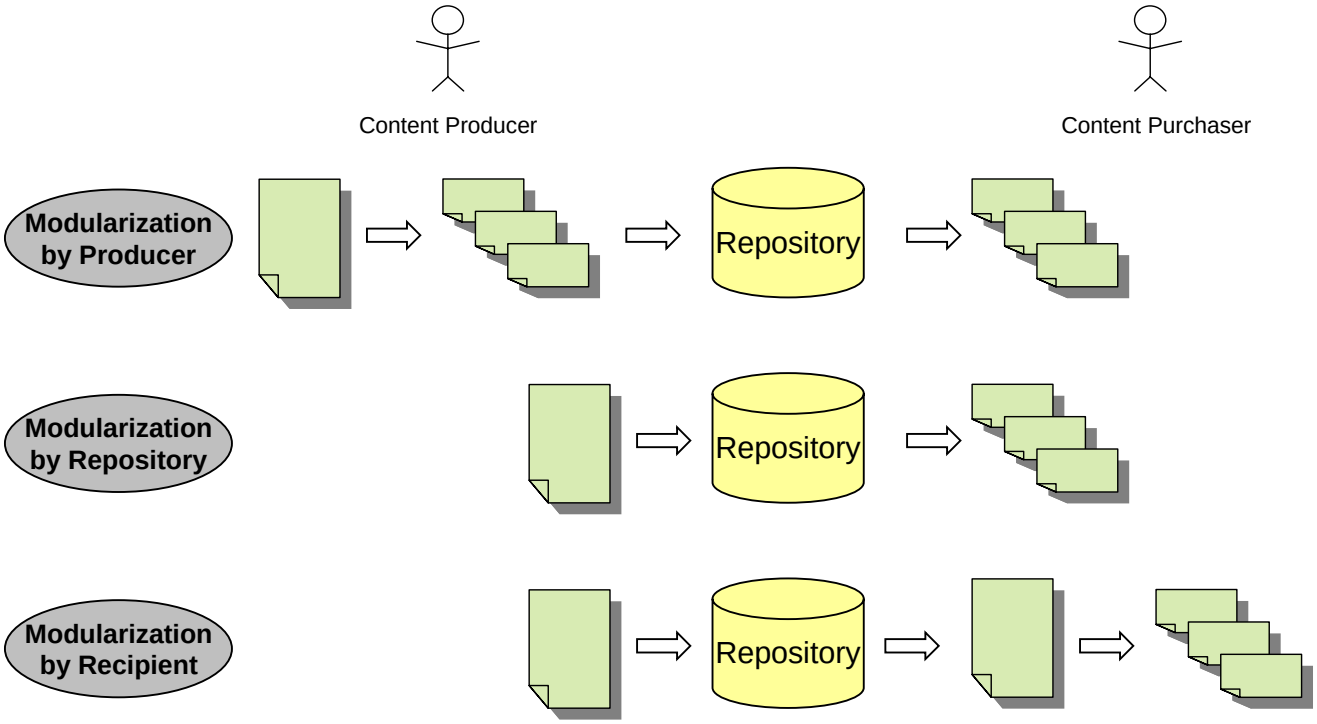


Figure 3.6: Modularization modes.

- The phase of the dissemination of a learning resource (before dissemination to a repository, during its storage in the repository, or after the retrieval by a re-user)
- The actor, who performs the modularization (author/owner, re-user, reauthor, third person, e. g. third party service)
- The intention of the modularization process (improving suitability for target educational context, increasing the sales volume, copyright issues)
- The degree of interaction with users

These aspects are linked to each other. For instance, the role of an author performs modularization before the dissemination, intending to increase his sales; maybe he is triggered by a potential customer, who is interested in fragments of a learning resource, but cannot perform the modularization himself. A re-user instead has in mind to maximize the suitability for his intended educational scenario and performs the modularization after the retrieval of a learning resource. These dependencies between the aspects have led to a reduction of generic modularization approaches to three archetypes. The archetypes – in the following also called *modularization modes* – are named by the actor as *modularization by producer*, *modularization by repository* and *modularization by recipient*. The three modes are listed in Table 3.1. For each modularization mode the table shows the dissemination phase in which the modularization takes place, the actor who performs the modularization and the possible degree of interactivity. Figure 3.6 illustrates the three modularization modes. A short description of all three modes is given below.

*Modularization by Producer* – In the first case, the content producer (author) modularizes the contents he has produced to allow recipients to reuse particular parts of his contents separately. The producer

Table 3.1: Modularization modes

| Modularization Mode | Dissemination Phase | Actor | Interactivity Mode |
|---|---|---|---|
| Modularization by Producer | Before upload to repository | Content Producer | Interactive |
| Modularization by Repository | At repository, after upload | System (LOR) | Automatic |
| Modularization by Recipient | After delivery | Reusing Author | Interactive |

retains control about which parts may be separately used and which not; he might even apply different licenses. This mode, though, does not regard the reusing author's needs; the recipient has no influence on granularity and number of target modules.

*Modularization by Repository* – If modularization takes place after a content producer has submitted a learning resource to the repository, the mode is called modularization by repository. The system is the actor and decides on what and how to modularize. Human users cannot directly influence the process, neither producers nor recipients. The only way to influence the modularization is to specify general algorithmic parameters for modularization. On the other hand, the achieved automation is potentially the most time efficient modularization method for the involved human actors. In addition, a standardized procedure may improve the overall quality and availability of modularized learning resources.

*Modularization by Recipient* – The third modus operandi is to let the reusing author modularize a learning resource according to his particular requirements. This approach allows the content recipient to extract and reuse exactly the contents he needs. However, this mode is more time-consuming for the recipient than to simply download already modularized learning resources. Drawbacks of this approach are that the user has to obtain (download and possibly pay) a large learning resource of which only a small part is of interest and that it becomes more difficult to find learning resources, which contain the desired contents.

In summary, all three modes make sense in particular scenarios. There is not one superior mode, but all modes have advantages and disadvantages for the involved human actors. The choice of a modularization mode depends on the particular interests of content producers and potential recipients and their influence on the decision. The following subsection describes a reference model for modularization processes that can be performed either by content producers before the dissemination of a learning resource or by content re-users after obtaining it. Afterwards, automatic modularization is discussed.

### 3.2.3 Requirements on a Process Model for Modularization

Most related works consider modularization only as a technical decomposition of well-specified elements [FMMF05, VJD+06]. This point of view neglects that the complete modularization process includes more tasks, for instance the finding and definition of the modularization goal, the decision making about the intended module boundaries, or the creation of metadata. Thus, technical decomposition is only one of several tasks that have to take place during the whole modularization process.

Doorten et al. [DGJ+04] have described a particular process for transforming classroom learning resources into digital learning resources. This description is closest to a modularization process of all known related work. However, it is a very specific approach that can hardly be applied for other scenar-

ios. But a generic process model for modularization processes would be a good foundation for designing and comparing different modularization processes. Starting with an analysis of the goals of process modeling, requirements on modularization processes are derived. These requirements lead to a generic modularization process model.

### Process Modeling

Humphrey and Feiler have defined a process as "a set of partially ordered steps intended to reach a goal" [FH92]. Splitting a learning resource into several smaller learning resources (modularization) could be such a goal; the activity of modularization may be called *modularization process* if it consists of partially ordered steps. According to Curtis et al. a *process model* is "an abstract description of an actual or proposed process that represents selected process elements that are considered important to the purpose of the model and can be enacted by a human or machine" [CKO92]. In other words, a process model is a generic blueprint from which tangible, more detailed processes can be derived. Curtis et al. name five basic uses of process modeling:

- Facilitate human understanding and communication
- Support process improvements
- Support process management
- Automate process guidance
- Automate execution support

Having a process model particularly allows to compare alternative actual processes that pursue the same goal and to improve the effectiveness and quality of processes. A generic model of modularization processes may help to compare existing modularization processes. Furthermore, the process model can serve as a foundation for improving existing modularization processes.

### Requirements for a Modularization Process Model

The creation of a generic modularization process strives for multiple goals. The first one is quality control: whether modularization is performed manually or automatically, the quality of modularization can be increased by sticking to a process model. The user is aware of the necessary steps and can keep the proposed order; furthermore, the execution of the individual process steps can be logged and provides an opportunity to be checked later for quality assurance. The generic modularization process can be concretized for particular formats, purposes and tools. A second goal is the development and optimization of modularization tools. Based on a modularization process a tool for supporting this process can be implemented. The existing tools mostly offer only a limited (merely technical decomposition) support of modularization. A process model can help to build tools that cover the whole modularization process. And finally modularization tools can be optimized by systematic analysis and improvement of process steps, which turn out to be either labor intensive or error-prone.

The basic requirement on a generic modularization process model is independence from used tools, formats, pedagogical methods, and also the technical realization of the process instances. The process has to be modeled in such a way, that the whole process and also each individual step can be implemented manually, automatically or semi-automatically. Manually means, that a user performs a task without technical support; automatic steps require no input from the user; semi-automatic implementations provide technical support, but rely on interaction with the user. Subject matter of the process is the modularization of learning resources according to the definition of Chapter 2. The process has to be suitable particular for digital Web based learning resources, such as learning resources complying with the SCORM specification. However, the process should be applicable also to other learning resources that are mainly based on text and static media. Static, continuous and interactive media (such as images, videos and flash animations) are considered as inseparable artifacts, which cannot be further modularized.

Modularization is regarded from the point of view of both content producer and content re-user. Especially re-users want to modularize, because they want to reuse only parts of a learning resource. Regarding the theoretical foundation, the reuse of only parts of a learning resource requires two modular operations – modularization and exclusion – to be performed. Thus, exclusion has to be kept in mind as a consecutive step. Even better is to include it optionally in the modularization process.

The generic process model is required to be pedagogically neutral. The process is not restricted to a particular pedagogical approach; however, it is assumed that each learning resource is based on a pedagogical content model – whether the content model is implicit or explicit[14]. Besides the content model, a learning resource always complies to a document model; in some cases, a finite number of document models are applied. A document model is a specification of the syntactical structure of a document. For instance, HTML, XML schemata or the Portable Document Format (PDF) specification are document models.

### 3.2.4 Review of Existing Modularization Approaches

The known existing modularization approaches have to be examined with regard to which tasks they involve and to what extend the requirements of a process model are already realized. Some approaches or tools, such as the Reload editor, allows the decomposition of learning resources, but this is only a partial aspect of the whole modularization. The ALOCoM framework provides plug-ins for Microsoft Office (MS PowerPoint, MS Word) for decomposing documents and presentations into smaller units (slides, paragraphs, media assets), and uploading them to a repository [VJD+06]. The extraction tool of Doan et al. performs automatic decomposition based on predefined extraction rules [DBD06].

The guideline of Doorten et al. focusses on the analysis of learning resources and how to determine good learning object boundaries [DGJ+04]. However, it is only a manual process, which has to be performed by subject matter experts. In terms of modular operations, the approach is a mix of modularization and exclusion. The core principle of this approach is to perform checks, analyzes and finally make decisions. Doorten et al. , too, express the need to clean up modules after the decomposition, e. g. by removing external references.

[14] A content model is explicit, if the learning resource format reflects the entities of the content model. An implicit content model is not reflected by the learning resource format, although the resulting learning resource structure might be reflected.

Rostanin et al. describe the SLEAM process [RL07], a process which supports workflow-embedded learning combined with the extraction and reuse of learning resources. SLEAM is a linear process, where each step has a clear outcome. Though this process is not a pure modularization process, it contains interesting elements. For example, the SLEAM process begins with the identification of a knowledge gap; this can be interpreted as the planning of a modularization goal. Two further steps of SLEAM, extraction and annotation are similar to decomposition and metadata creation.

There is a reference model for quality management of planing, development, execution and evaluation of educational processes, called PAS1032-1 [Ham04]. However, this reference model is too generic; it does not contain relevant process steps that occur in modularization. PAS 1032-1 was originally intended for the description and documentation of processes for creating new learning resources or learning services from scratch. However, with some adjustments, the specification could also be interpreted for the creation of modular learning resources out of existing ones.

A reasonable approach for a generic modularization process is a combination of the approaches of Doorten et al. , Rostanin et al. and the PAS 1032-1. The universality claim and flexibility of the DIN PAS 1032-1 should be combined with the thorough analysis and decision making of Doorten et al. . From Rostanin, the process orientation and the alignment with a user's goals can be adopted.

The review of existing modularization approaches disclosed that a lot of different tasks could be involved in modularization. Some of these tasks appear in the different approaches in a very similar form. For example, the definition of an intended goal is frequently mentioned; also the analysis and decomposition is often relevant. Other tasks are required only in particular settings. For instance, a separation of content and activities is performed only by Doorten. A modularization process may comprise different tasks. The quality of the modularization result is influenced mainly not by the decomposition, but by the accompanying tasks. Potential tasks that have to be obtained from the above review of existing modularization approaches are e. g.

- Planning and goal definition: The user specifies on a high level which contents he wants to reuse and what for [Ham04, RL07].
- Analysis of the learning resource: The user has to get to know the contents of the learning resource. Either he views and assesses the learning resource manually, or he is provided a summary of the contents by a supporting tool [DGJ+04].
- Decision making: Knowing the contents of the learning resource and a modularization goal, the user decides the intended module boundaries [DGJ+04, Ham04].
- Decomposition: The technical decomposition is the realization of previously determined module boundaries. The contents are distributed to multiple separate modules [VJD+06, DBD06, RL07].
- Error detection and correction: Modularization might lead to errors in the resulting modular structure. Errors can be technical defects, such as void references, or errors regarding the contents. Particularly a combination of modularization and exclusion can easily lead to broken textual references to other parts of the original learning resource, which are no longer guaranteed to exist [DGJ+04].

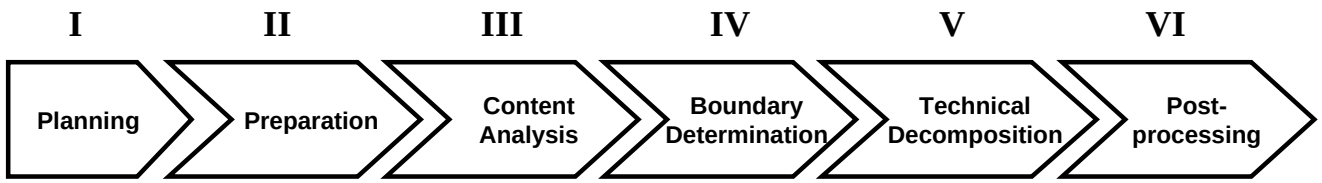


Figure 3.7: Interactive modularization steps.

- Metadata generation: Each new module requires an individual metadata record. The generation of metadata is considered to be integral part of the modularization. Metadata generation should be performed along with the modularization, because users otherwise tend to forget the creation of metadata [RL07, VJD+06].

Based on these observed tasks, a generic modularization process model can be constructed. This generic process model conceptually covers the whole modularization process from goal definition to final post-processing tasks. The process model should not describe concrete tasks to be performed, but abstract process steps, which serve as categories covering different potential tasks.

### 3.2.5 A Generic Reference Model for Modularization Processes

The above requirements and identified modularization tasks have resulted in a generic process model for the modularization of learning resources (see Figure 3.7). The process model is designed as a linear sequence of six process steps. A linear order facilitates comparisons of different processes. Furthermore it ensures repeatability of results if process steps are always executed in the same order. Besides rearrangements of process steps, it may also happen that particular steps are omitted in implementations. The given order has been chosen because of potential dependencies between the involved tasks. For example, it is hardly possible to determine module boundaries without having analyzed the learning resource before. If the order of process steps in a process instance differs from the reference process model, the reasons for the change should be documented.

Each process step stands for different tasks that may be performed in that step, depending on the particular situation. There is no a priori list of potential tasks; the tasks to perform have to be analyzed and implemented depending on the environment and parameters of the modularization. The existing modularization approaches, which have been reviewed above, can be expressed as instances of the generic process model. For this purpose, the approaches have to be decomposed into tasks that each fit into one of the process steps.

#### Process Steps of the Modularization Process Model

A modularization process starts with the planning of the modularization and ends with finalizing post-processing tasks. The six process steps (see Figure 3.7) of the process model are specified as follows:

I – Planning. In the planning phase, the modularization goal has to be specified and methods and criteria to apply have to be identified. This includes the intended re-use purpose for the

resulting modules (simple reuse or aggregation; relevant properties of target group and scenario) and also the question of which criteria to apply (contents, didactic and media criteria) and which module granularity to chose.

II – Preparation. Before modularization can take place, preparation may be required. If a learning resource exists in an arbitrary format, it might be desired to transform it into a format that is better suited for re-use. Implicit course structures may be made explicit and can then be used for determining suitable fragments.

III – Content analysis. In a further step, the learning resource and its constituting elements are analyzed regarding different criteria, which may have an impact on the determination of reusable fragments. The analysis comprises properties of the contents of fragments, their pedagogic functions and media properties (e. g. usage of particular media formats). Content properties, for example, may be topics covered, similarity of fragments or references between fragments. Pedagogical properties are e. g. the didactic function of a section, such as *theory*, *example* or overview knowledge. Information about the involved document and media formats can provide valuable hints for the determination of module boundaries.

IV – Boundary determination. Based on the information, which has been collected in the previous process steps, the fragments that are to be transformed into modules are chosen. Boundaries for the resulting modules are determined by either the user or an algorithm. Usually, resulting modules are supposed to be consistent and self-contained; but depending on the reuse purpose, a deviation from this principle can be acceptable .

Two different modular operations can occur: modularization and exclusion; though these two operations are supposed to appear combined in practice. Modularization is a partitioning of a learning resource into several modules, whereas exclusion decides which fragments are removed from the overall learning resource and which are not. That is, exclusion may result in only one target module. In contrast, each fragment of a learning resource is assigned to one of the resulting modules by a pure modularization method. In the case that no exclusion occurs, the boundary determination is also called segmentation.

V – Technical decomposition. The decomposition step realizes the determined module boundaries by splitting the learning resource. The resulting modules are made compatible to a predefined modular learning resource format, which is suitable for the intended use. If, for example, the result is supposed to be integrated into a larger course, the module format should allow the aggregation of modules. Contents – snippets or whole files – are distributed to the to-be modules; contents should become redundant only in those cases, where this is necessary to ensure the consistency of the new modules.

VI – Post-processing. The results of decomposition are very often not yet completely suitable for re-use. There are still references from one content fragment to another one, which no longer exist in the same module, contents have become inconsistent, and old metadata records do no longer correctly describe the new modules. These shortcomings have to be eliminated in

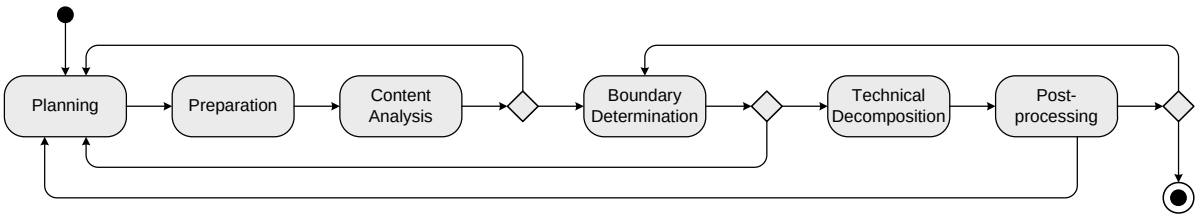


Figure 3.8: Modularization process including feedback cycles.

a post-processing step. Not all of these tasks can be automated - some require a manual intervention of the user.

The whole process, but also each individual process step can be implemented as a manual, semi-automatic or automatic tool. A manual implementation leaves the execution of the tasks completely to the user; it gives only a guideline what and how to do. An automatic implementation provides a tool, which performs the process or process step without interaction with the user. Though, it is possible to define general parameters that control the behavior of the tool. Semi-automatic implementations support the user; the user is guided through the whole process, but is enabled to interactively control the process flow, the executed tasks and their parameters.

In practice, not all process steps have to be implemented. But it is recommended to arrange at least a manual check of all process steps to ensure that the user is aware of necessary manual tasks. Ideally, the modularization process gets along without iterations. However, implementations can introduce feedback cycles in order to allow a revision of previously made decisions. If decisions are changed, subsequent steps potentially have to be repeated to avoid inconsistencies. Figure 3.8 shows a modularization process with feedback cycles.

### Application Example: Modularizing SCORM

The generic process model has to be instantiated as a particular modularization process that suits the general conditions and goals of the scenario. Different considerations and parameters influence how the process steps are to be implemented. For the reuse scenario of Chapter 2 and SCORM as a learning resource format some exemplary considerations are presented here.

How can the generic modularization process be applied to the modularization of SCORM based learning resources? The implementation of a SCORM modularization depends on the intended module granularity. Three cases can be differentiated:

- The module boundaries align with the existing organizational structure of the SCORM package.
- The module boundaries cut the smallest elements of the organizational SCORM structure.
- Media assets are desired to be reused as separate modules.

The first case can be handled in a straightforward way. The user views the package structure and assigns structural elements to new modules. In case of exclusion, he may also decide which elements to remove. Decomposition is performed by moving SCORM elements with all recursive dependencies to a

newly created module. Metadata for the new module is either automatically generated or entered by the user.

The second case, modules boundaries cutting elements, is harder to cope with. The internal structure of a structural element (typically a SCO), has to be analyzed and presented to the user. In the decomposition phase, the contents have to be separated, and internal navigation elements of the SCO have to be adapted. It is also possible to split SCOs already in the preparation phase into multiple assets. The resulting structure is more likely to be modularized according to the first case.

In the case of reusable media assets, the SCORM structure is less relevant. A possible realization of the modularization process is to identify all media resources (images, audio, video, interactive media) and present them as a sequential list to the user. Attributes and metadata of each resource can be used for sorting and filtering the list. A preview function could also facilitate the decision-making.

Two implementations of a modularization process for SCORM learning resources are described in Chapter 7. The implementation and experiences gathered in the Content Sharing[15] project are discussed and compared in that chapter in detail.

### 3.2.6 Automatic Modularization

The generic modularization process is mainly intended for interactive modularization, where the user can influence the resulting modules. But interactive modularization is not always the best choice. In some cases, fully automatic modularization is required. Interactive modularization requires a user to perform the time-intensive modularization and to intentionally define a modularization goal. The two interactive modularization modes are modularization by producer and modularization by recipient. The producer often does not want to spend more time than necessary. But on the other hand, recipients often wish to retrieve already modularized contents. Furthermore, small modules might be easier to retrieve than a large learning resource containing only a small fragment of interest. Considering the trade-off between effort and resulting module quality (in terms of congruence of reuse goal and module boundaries), automatic modularization may become the favored mode.

Another reason for automatic modularization is that interactive modularization is a bottleneck for availability. Producers often do not know which parts of a large learning resource are desired for reuse by other authors or teachers. Also, even if an author selects some potentially reusable parts and makes them separately available, there may still be more parts that could be reused later.

In consequence, a large number of fragments of a learning resource are potentially reusable. As described in Chapter 2, learning resources are often hierarchically organized with regard to a content model. Availability can be maximized by making all fragments of a learning resource at each aggregation level of a content model separately available. This task no longer requires the interaction of a person; this kind of modularization can be performed automatically. Ideally, this full modularization is performed within a repository immediately after a learning resource has been uploaded. A drawback of generating a maximum number of modules is that retrieval performance could suffer because of the increased total number of learning resources in a repository.

[15] http://www.contentsharing.com

To implement automatic modularization completely, the generic modularization process can also serve as a foundation. The planning step has to happen in advance, by defining system parameters that apply to the modularization of all provisioned learning resources. Preparation, content analysis and the determination of module boundaries happen completely automatically based on the predefined parameters and methods. Technical decomposition is the same as in the interactive process, as it requires only the determined boundaries as input. Post-processing also needs to be performed automatically, though the quality of completely automatic post-processing is supposed to fall behind a human quality check. When modularization is performed in an automatic way, there is no user, who may enter metadata for all the new, small modules. Hence it is necessary to automatically generate metadata. Automatic metadata generation is addressed in Chapter 6.

## 3.3 Summary

Two contributions have been made by this chapter: a module concept for learning resources as a foundation of modular learning resources; and a generic modularization process as one of the key modular operations on and towards modular learning resources. The module concept builds upon the existing SCORM specification, keeps compatibility for existing SCORM-compliant systems, and therefore increases the modularity of SCORM-based learning resources. It is now possible to modularize and aggregate SCORM packages beyond the traditional *copy&paste* approach. Having this module concept available, modularization – the process of splitting a learning resource into multiple modules – was researched. Some approaches for learning resource modularization (also known as decomposition) can be found in literature. As a synthesis of the aspects of known decomposition approaches, a generic reference process model for modularization of learning resources has been developed. This reference model covers the whole modularization process including goal specification, decision-making, as well as pre- and post-processing. A proof-of-concept implementation of the module concept and the modularization process in form of a modular SCORM editor combined with a re-purposing tool – enabling modularization, aggregation and adaptation of learning resources – is presented in Chapter 7.

# 4 Aggregation of Modular Learning Resources & Retrieval for Aggregation

There are six modular operations on learning resources. Modularization as one of these operations has already been covered in the previous chapter. This chapter now investigates aggregation as a further modular operation. Aggregation has been defined in Chapter 2 as *the process of combining small learning resources into a larger learning resource.* In our example from Subsection 2.3.1 Sarah wants to aggregate several existing learning resources in order to create a new learning resource on organizing a workshop.

There are many examples of tools in the area of e-learning, which enable users to aggregate learning resources. For instance, Sanchez and Sicilia describe a system that automatically aggregates learning resources, which in combination help to achieve a given learning objective [SS04]. Aggregation is also strongly related to content models (cf. Chapter 2). Content models describe the different types of learning resources and how they may be validly aggregated. A particular type of aggregation is *authoring by aggregation* – an authoring paradigm founded on aggregation as main construction operation.

The first section of this chapter reviews existing authoring by aggregation systems. An extension to these authoring by aggregation systems is proposed in Section 4.2. Finally, Section 4.3 presents improvements of retrieval for aggregation. This chapter contributes to the overall goal of the thesis by improving availability and aggregation, as illustrated in Figure 4.1.

## 4.1 Authoring by Aggregation Systems – Related Work and a Scenario

The production of learning resources is often based on didactic guidelines, following well-known best practices. Especially large content producers rely on content models as blueprints for didactically well-structured courses. Some of these content models have been presented in Chapter 2.

When learning resources are assumed to be structured, authoring tools can and should also support a structured authoring process. There are different kinds of authoring processes, depending on which and how many learning resources are produced, which and how many roles are involved and last but

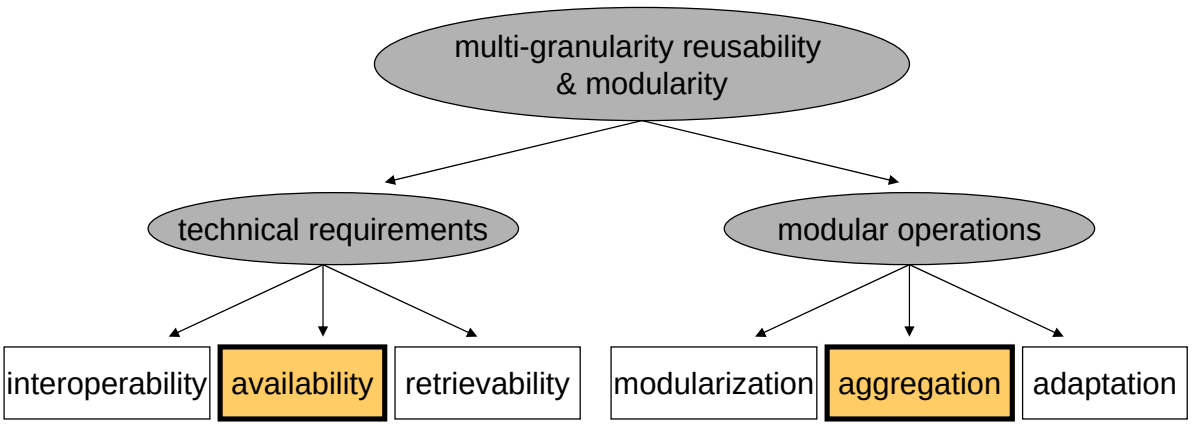


Figure 4.1: Contributions of Chapter 4 to multi-granularity reusability.


Figure 4.2: ADDIE process [NHHM+04].

not least which tools are used. Authoring by aggregation is one of these authoring processes. As defined by Duval and Hodgins authoring by aggregation means that learning resources "are created by selecting content/information objects from a repository, usually with the significant assistance of metadata and profiles to do so." These content objects (which are also learning resources according to the definition of this thesis) are assembled to form a new learning resource [DH03].

### 4.1.1 Review of Existing Authoring by Aggregation Systems

Production of learning resources can be organized and performed in many different ways. The different approaches differ in various aspects, depending on the intended use. Some of these differing aspects are:

- Single vs. multi user system (number of roles)
- Number of separate consecutive production phases (separation of concerns)
- Support of didactic design
- Reuse of existing contents at different levels of granularity
- Access to local vs. remote and homogeneous vs. heterogeneous repositories
- Support for adaptation of reused contents
- Integration of third party tools

Instructional design often follows the ADDIE model [NHHM+04]. ADDIE stands for five developments phases for the development of learning resources; the phases are *analysis*, *design*, *development*, *implementation* and *evaluation* (see Figure 4.2). The ADDIE model has evolved over time and has become commonly accepted as design principle [Mol03].

Authoring by aggregation is an authoring paradigm that is primarily based on reuse of existing contents. The term *authoring by aggregation* has been mentioned first by Duval and Hodgins [DH03]. It has been realized as an authoring system by Hoermann in the ResourceCenter [HHRS05]. The ResourceCenter is an integrated repository and authoring tool, which is used mainly by university lecturers for creating Web based trainings. The ResourceCenter has a proprietary content format and allows to export courses into the SCORM format. A second system supporting authoring by aggregation is the ALOCoM framework [VJD+06]. ALOCoM facilitates the reuse of Microsoft PowerPoint slides and slide fragments. Slides and paragraphs are stored in a repository and can easily be added to a new slide set by a PowerPoint plug-in. A second add-in for Microsoft Word has also been published recently.

Hoermann has designed two versions of an authoring process for his system [Hoe05]. The first process is an abstract process that describes the basic functionality of the authoring of a learning resource. This

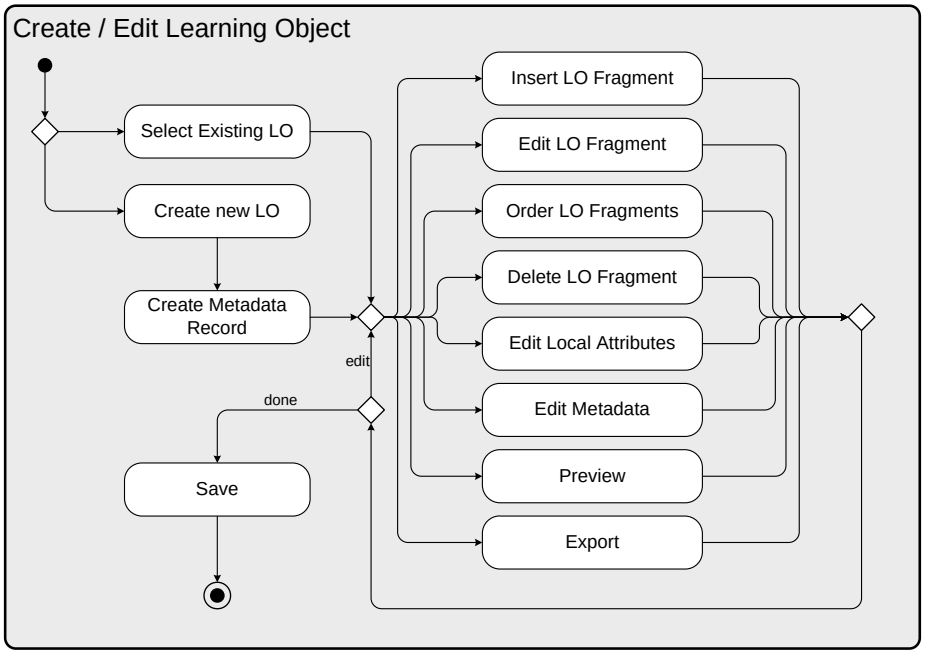


Figure 4.3: Extended authoring process as defined by Hoermann [Hoe05].

process serves as the theoretical foundation of the system. The extended authoring process relaxes the order in which activities are performed. This extended process has been reported to be more suitable in practice. The process is illustrated in Figure 4.3.

### 4.1.2 A Scenario for Authoring by Aggregation

According to the scenario described in Section 2.3.1 a user called Sarah has to create a learning resource on workshop organization. She wants to reuse existing learning resources that are stored in different repositories. It can be assumed for now that the learning resources have already been modularized. Thus, the reusable contents are all available at the proper granularity.

Sarah is a semiprofessional author – authoring of learning resources is not her primary profession. She is subject matter expert, author, instructional designer, content developer and media developer in a single person. She does not have to coordinate her work with other persons; there is no need for a documentation of the learning resource development; implementation and evaluation of the learning resource are also out of scope of the original development. The author has a rough idea in mind of what the new learning resource should be about and what the structure of such a learning resource could be.

How would Sarah perform the authoring using one of the existing authoring by aggregation systems? She starts with an idea in her mind about the structure of the learning resource; an analysis phase according to the ADDIE model has already happened informally before. Perhaps she even sketches the structure of the learning resource on a sheet of paper, taking some notes about topics that might already exist as reusable learning resources. The sketching and note taking matches the ADDIE design phase. Support of current authoring by aggregation systems start only after the first to development phases have passed.

Now, Sarah accesses various repositories in order to search for learning resources that might be suitable for aggregation. She has to use the search functionality of different repositories for finding suitable learning resources. As she does not know which learning resources are available in advance, she has to search for all partial topics of her course that she has on her list. Apparently, searching for a large number of modules might become very time consuming.

After some time, Sarah has found three reusable learning resources: a guideline for publishing workshop proceedings, and two parts of a university service handbook covering room booking and catering services on her campus. She is also now aware that no other reusable learning resources for her purpose exist. Sarah downloads the three learning resources and imports them into her authoring by aggregation system for aggregation.

As this use case demonstrates existing authoring by aggregation systems support only the development phase of in the narrow sense. The previous phases of analysis and design are not reflected by the functionality of authoring by aggregation tools. There is potential for improving authoring by aggregation by also supporting these tasks.

### 4.1.3 Challenges for Authoring by Aggregation

There are two challenges that this chapter addresses: the unobtrusive integration of further development phases, and the improvement of retrieval for aggregation.

Authoring by aggregation is a light-weight authoring approach that is particulary based on the selection and aggregation of existing learning resources. One characteristic of existing authoring by aggregation approaches is that they are narrowed to the development phase of the ADDIE model. Particularly, extensive instructional design tasks are left out in order to achieve lean tools.

However, including technical support for the first ADDIE phases – analysis and design – could improve authoring by aggregation. The support should be unobtrusive and optional in order to retain the light-weight characteristic of the authoring by aggregation approach. Thus it is a challenge to integrate analysis and instructional design in an unobtrusive way into authoring by aggregation processes.

There are other professional authoring systems and processes, which allow sophisticated planning of content production, providing multiple roles[16], different production phases, resource planning and scheduling. One of these systems is currently developed in the Explain project [ZBC+05]. However, these professional systems are an overkill for occasional authors. What is needed is a combination of the simplicity of authoring by aggregation tools and the support of different development phases provided by professional authoring systems.

Another shortcoming of existing authoring by aggregation systems is that the search for existing learning resources in repositories does not utilize information that may be available from the analysis and design phases. In the analysis phase of learning resource development, conditions and parameters of a new learning resource are specified – for instance information about the target learning environment and target groups. The design phase specifies additional information of content elements of the learning resource. For instance, the granularity, didactic purpose and hints on the topics could be present for each planned content element. This information could be utilized for retrieval.

[16] A general role concept for the authoring of multimedia content has been developed by Liepert [Lie01].

## 4.2 An Extension to Existing Authoring by Aggregation Processes

There are two challenges that were identified for authoring by aggregation. The first challenge is to integrate further development phases (particularly the design phase) into authoring by aggregation systems. And second, information from the analysis and design phases should be utilized for retrieval of learning resources. This section introduces an extended authoring by aggregation process that takes the need for a separate design phase into account.

The authoring by aggregation process presented in this section has to be understood as an evolution of a process rather than the introduction of a completely new process. Duval and Hodgins have laid the foundation for authoring by aggregation in the LOM Research Agenda [DH03]. They describe the concept of authoring by aggregation as the creation of a learning resource by selection and composition of existing learning resources from a repository. The LOM Research Agenda does not mention whether (and how) selection of existing learning resources and creation of missing contents coexist. Hoermann has recognized that although the reuse of existing learning resources is the foundation of authoring by aggregation, there are most often still gaps that have to be filled with new contents. Consequently, Hoermann's authoring by aggregation process lets the author choose between selecting an existing learning resource and creating a new one for each position until the learning resource is complete. The concept of parallelizing reuse and creation has proved to be reasonable. Hoermann's process requires that the author creates the structure of a learning resource and the contents at the same time. But this concurrency prevents the author to thoroughly design the didactic structure of the learning resource. This shortcoming is eliminated by an improved authoring by aggregation process: A separate didactic design phase is scheduled to take place before the content authoring.

Also, the user does not first have to choose whether he wants to reuse an existing module or create a new one – in the didactic design phase the author creates only a placeholder, afterwards he tries to retrieve a suitable existing module for this placeholder; if no suitable module exists the placeholder remains empty and has to be filled with new contents later. A placeholder is described with various attributes that indicate intended instructional and topical properties. These attributes are available in the retrieval phase as additional information.

The improved process also contains a further phase for adaptation of included modules. The need for this adaptation phase originates from the heterogeneous scenario of this thesis: when modules from different sources are combined, the mosaic effect occurs and has to be corrected. The mosaic effect and adaptations are dealt with in more detail in Section 5.1. A further improvement to be introduced is the stronger integration of retrieval into the authoring by aggregation process. The Hoermann process, for example, assumes that all modules are already available in the authoring environment. Learning resources from different repositories have to be retrieved and imported first in the authoring environment. The new authoring process instead assumes that the potentially useful modules are possibly spread over several heterogeneous repositories. It is a function of the authoring process to provide access to these repositories and let the user directly retrieve and reuse modules from these repositories. It should be no longer necessary to switch between the authoring environment and retrieval interfaces of repositories.


Figure 4.4: Improved strict authoring by aggregation process.

### 4.2.1 Authoring Phases of the Authoring by Aggregation Process

In total there are five authoring phases: the didactic design phase, the retrieval & replacement phase, the adaptation phase, the content creation phase and the publishing phase. These phases are described as follows.

- **Didactic design phase.** In this phase the author plans the structure of the learning resource. According to the structure of SCORM courses, the structure of learning resources is designed as a tree. It consists of placeholder elements, which will be replaced by existing modules or new content later. Each placeholder element has to be described in this phase to support the subsequent phases. An element is specified by a brief description of its contents (keywords or short phrases) and didactic attributes according to the underlying didactic content model (e.g. from a didactic ontology [Med00]). Didactic attributes are especially the intended granularity and - where applicable - a didactic type, such as 'introduction', 'theorem' or 'exercise'.

- **Retrieval & replacement phase.** After the structure has been designed, the attached repositories are searched for suitable existing learning resources. The specified attributes of a placeholder element are used as search terms. Additionally, further information (cf. Section 4.3) may be used for optimizing the queries. A list of suitable learning resources for each element is presented to the author. He may either select one of these learning resources for reuse or mark the placeholder for production of new contents. If the list contains a large number of learning resources it has to be ranked appropriately.

- **Adaptation phase.** If contents from different authors and possibly created by different authoring tools are aggregated, the result might look like patchwork in the first place. It is necessary to unify the contents. In the adaptation phase contents are transformed into a unified representation. A detailed description of content adaptation methods can be found in [ZRS06, ZBRS06].

- **Content authoring phase.** Typically not all placeholders may be replaced by existing contents. Some placeholder elements remain even after the replacement phase. For these elements, new content has to be created by the author. In contrast to the Hoermann process, no recursion takes place, as the complete structure has already been specified in the first phase. External editors may be integrated for creating and editing the content.

- **Publishing phase.** In the final phase the new learning resource is published. The publishing phase includes creating a metadata record for the learning resource and transferring it to a repository from where it can be used and reused.

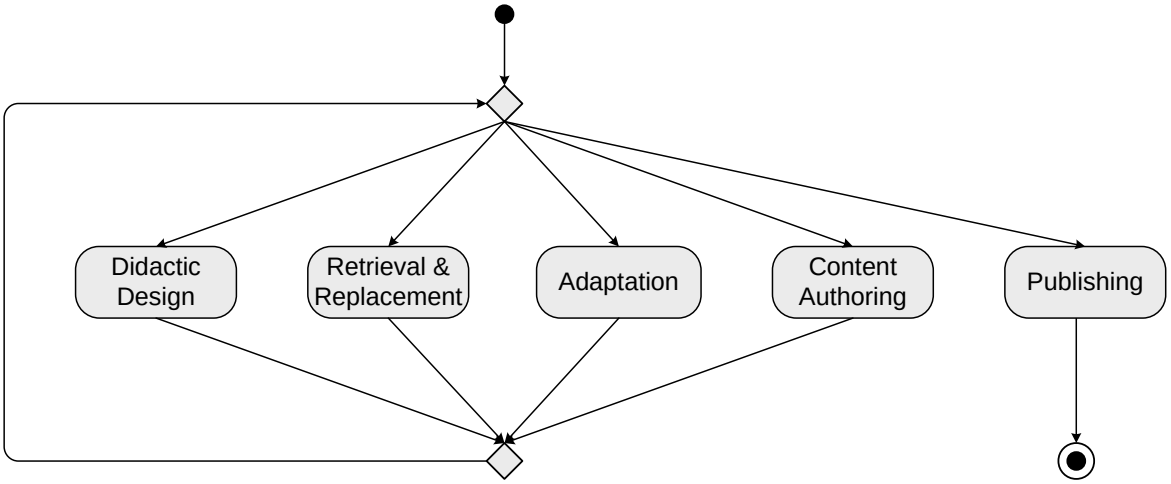


Figure 4.5: Relaxed authoring by aggregation process.

### 4.2.2 Strict and Relaxed Process Implementation

The described phases may be arranged in a strictly sequential way. Each phase has to be completed before the author may proceed to the next one. Fig. 4.4 illustrates the strict authoring process. However, the strict implementation is mainly intended as a theoretical concept.

In practice, the author will not always be able to finish the phases one after another. Imagine that an author has designed a course structure, replaced most placeholders, adapted the contents and is now creating the missing contents. Suddenly he realizes that an exercise is missing in his structure. The strict process would not allow to go back and change the structure. Therefore we define a relaxed process, which allows iterations of authoring phases. In the relaxed process, the first four phases may be repeated in any order. The author may choose at any time to either edit the course structure, to replace placeholder items with existing learning resources, to adapt a learning resource or to create new content. When the author has decided that the learning resource is finished he invokes the publishing phase and thereby ends the process. The relaxed process is shown in Figure 4.5.

## 4.3 Improving Retrieval for Aggregation

One of the phases of the introduced authoring by aggregation process is retrieval of reusable learning resources. Particularly when a lot of learning resources need to be found one after the other, the success of reuse depends on effective and efficient retrieval methods. This section analyzes how retrieval in LORs has been performed in the past and which recent approaches have emerged. Known approaches from information retrieval and multimedia information retrieval are taken into consideration for the application in LORs. An improvement of LR retrieval based on the usage of aggregation context information is introduced.

### 4.3.1 Retrieval of Learning Resources for Aggregation

Retrieval has been a research topic long before it became relevant in the area of e-learning. Information retrieval deals mainly with search and clustering of text documents whereas multimedia information retrieval also takes into account non-textual attributes of multimedia objects [Sch05]. Retrieval of learning resources can be considered as a special case of multimedia information retrieval. Hence, many sophisticated concepts for retrieval of multimedia objects could be transferred from multimedia information retrieval to retrieval of learning resources.

Existing repositories usually provide a retrieval interface in one of two modes: either a pure keyword search, or a complex form, which allows to specify values for several metadata fields to match. Besides graphical user interfaces (e. g. via a Web form) there are also query protocols that enable to query a remote LOR. An example for such a protocol is the Simple Query Interface (SQI) [SMvA+05]. SQI does not provide a query language but only specifies how a query and the results are transmitted. The most commonly used query language in combination with SQI is the Very Simple Query Language (VSQL), a query language that restricts queries to a set of keywords. With the ProLearn Query Language (PLQL), a new query language for learning object repositories has recently been developed [TMC+08]. PLQL supports different query levels with increasing complexity. Up to now, three levels (0 to 2) have been specified; more complex query levels are yet to come. PLQL query level 2 supports hierarchical metadata specifications (such as LOM) and also range comparisons ($>, <, >=, <=$). However, queries of level 2 still generate only a separation of all learning resources into two sets: learning resources that match the query and learning resources that do not. PLQL does not specify how ranking of the result list is performed.

Parallel to solutions presented in this thesis, Ochoa et al. have recently introduced ranking metrics for learning resource queries [OD07]: several ranking metrics are given, which are based on usage relations between learning resources, on the history of learning resource usage by the querying user or similar users, and on the context a learning resource is intended for. Ochoa builds upon known ranking methods from general information retrieval and interprets them for the particular requirements of learning resource retrieval. An evaluation has led to the conclusion that a linear combination of several metrics is best suited for practical application. A prerequisite for the application of Ochoa's metrics is that usage relations between learning resources are tracked and are available at the repository for rank calculations. A system for the tracking of usage and reuse relations is, for instance, the Lifecycle Information System developed by Lehmann et al. [LHRS07].

One focus of information retrieval research, which has gained importance over the last years, is query expansion [MSB98]. Query expansion is a method of adding further search terms to a query to enhance the effectiveness – precision, recall or both – of query results. The additional query components may be used to either filter the result set, to enlarge it or to change the ranking of the result set. Recently, a new framework for learning resource retrieval has been developed by Dolog et al. that supports query rewriting and ranking-based recommendations [DSNK08].

While information retrieval deals mainly with pure text, multimedia objects have different attributes that may be relevant for a query. Therefore, complex multi-attribute and similarity-based query languages have been developed. An example for such a query language is WS-QBE [SSH05]. Some of the features

of the language, such as usage of similarity functions or preference values could be very useful in the area of LOR queries, too.

Relevance feedback systems are used for iterative optimization of queries based on the user's evaluation of query results. The system presents query results to the user, who then may give feedback about the quality of each result element. In practice, relevance feedback methods often generate additional search terms for query expansion.

Concepts for retrieval systems, such as ranking, fuzzy search, query expansion or relevance feedback are still not commonly used in learning object repositories, even if the technologies have been available for years in other research areas. It should be one goal of future e-learning research to leverage this potential.

Based on existing methods from (multimedia) information retrieval, some new approaches for improving learning resource retrieval could be:

(a) Complex query languages, which allow to combine several LOM fields. A query language should also support similarity-based search, as well as preference values for individual components of a query. Distance or similarity functions have to be specified for the various different LOM fields and their possible values to overcome the limitation to binary comparisons.

(b) An alternative approach for similarity-based search would be to use adaptation effort as measure. Known approaches for similarity functions only compute how much a LOM record deviates from a query. The alternative measure would estimate the adaptation effort that is required to transform a particular learning resource into one, which fully complies to the query.

(c) The context of an authoring environment could be used as input for query expansion methods. Especially metadata from already loaded learning resources may be used to search for learning resources, which better harmonize with the existing ones.

The last two mentioned approaches, utilizing adaptation effort estimations and aggregation context information for improving retrieval, will be discussed further in this section.

### 4.3.2 Retrieval Based on Aggregation Context

As mentioned in the previous subsection retrieval methods for learning object repositories still lack some functionality. This section focusses on how the aggregation context of an authoring process can be used for query refinement.

Search result lists of LOM queries may become quite large. Especially if a user searches for multiple learning resources one after another, the required time adds to the production costs and may make reuse of learning resources inefficient. Query expansion, which has been explained in the previous subsection, could improve the quality of retrieval results and thereby reestablish the economical benefit of reuse.

#### Context and Aggregation Context

Context information from authoring environments and relevance feedback mechanisms are promising candidates for query expansion. There are various notions and definitions of what *context* is [GÖ5,

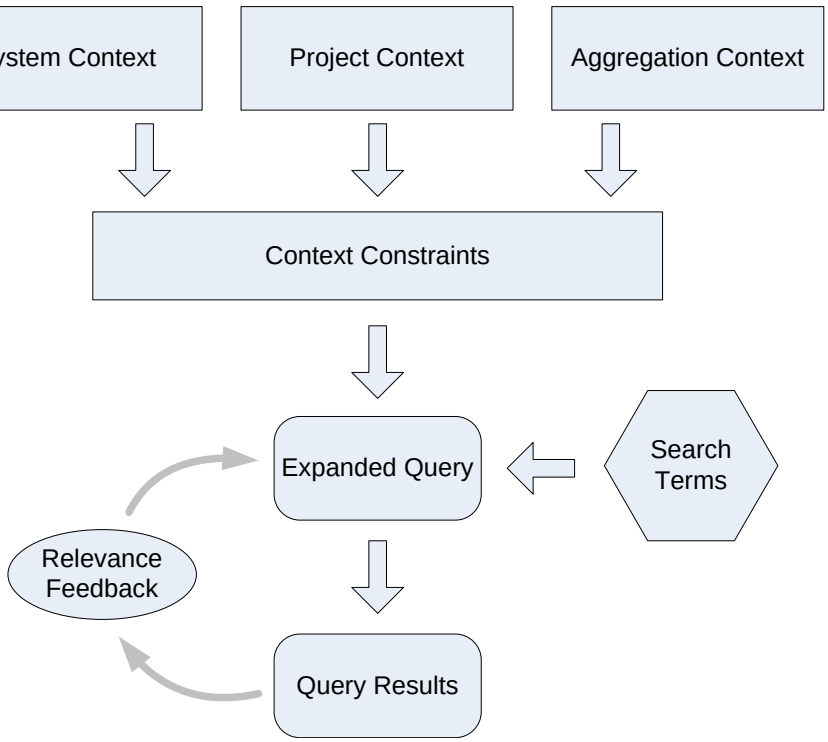


Figure 4.6: Usage of context information as input for query expansion.

Wes05]. Context is understood in this thesis as any information that is known about the author, his authoring environment and the tasks he is performing. Analog to the definition of Görtz [GÖ5], a context consists of a large number of context attributes. We assume that the metadata record of a learning resource can be considered as an approximation of context; metadata fields are context attributes in that sense. Different types of context information are imaginable, particularly system context, explicit project context and aggregation context (see Fig. 4.6).

As **system context** we subsume all information about the user, the tools and systems he uses and what he has generally done in the past. Exemplary system context information is for instance technological restrictions of his authoring system (e.g. which image and video formats are supported).

**Explicit project context** is all information, which has been explicitly specified by the author about the project he is currently working on. Project here typically refers to a particular learning resource the author is working on. Examples for project context are the intended course language, target document formats, target group (age, role, difficulty level, interaction level, etc.). Also, limits for the total amount of learning time or acquisition costs may have an impact on which learning resources are suitable for aggregation. This information may be the written outcome of an analysis phase of learning resource development.

**Aggregation context** – which may also be called implicit project context – is information about the current project, which is deduced from the contents already existing in the project. Aggregation context information may be deduced either from the contents itself or their metadata records; in practice, metadata will be easier to use. The more learning resources are aggregated in a project over time, the better the aggregation context may be automatically determined.

Which metadata fields are suited for determining the aggregation context? Especially those fields that are typically the same for the whole assembled learning resource; for instance the language, coverage, target group, presentation form or technical requirements are suited. In some cases the origin (author, catalog, etc.) of a learning resource or its classification might also give valuable hints. Besides the significance of a value as such, the format of a metadata field is also an important criterion. Metadata fields are only useful as context source if syntax and semantics are clearly specified and can be compared. Consider as example that one learning resource has a coverage specified as "16th century France" and another one is described as "Paris, 1537". A human could easily recognize the second description as subset of the first one; achieving an automatic matching by algorithms, though, would be very tricky.

Context information of the three types can be merged into combined context constraints. These context constraints are then used for query expansion, either for filtering (removing of result entries) or for improving the result ranking. Additionally, relevance feedback may be used to rewrite queries according to a user's feedback regarding previous query results.

A similar approach has already been proposed by Sanchez and Sicilia [SS04]. They apply the *design by contract* paradigm from object-oriented software construction to the composition of learning resources. Their goal is to automatically select and aggregate learning resources for a given learning objective. Beside other constraints they also use aggregation constraints to select suitable learning resources. However, because of the logic-based design-by-contract paradigm the approach supports only a binary matching: either a learning resource fits or it does not. Furthermore, a formal contract has to be specified for each learning resource; thus, the method is not applicable to conventional LOM records.

In reality, learning resources – and their metadata records – are rarely perfectly fitting. But a near enough match is often still satisfying. Moreover, learning resources can be adapted to better fit into the new course [ZRS06]. Therefore, fuzzy queries, which produce a ranked result set, are better suited than strict matching. Otherwise, many relevant objects would be missing in the result sets. For the rest of this paper it is assumed that query expansion affects only the ranking of learning resources but does not exclude learning resources from the result set. Query expansion is achieved by determining the aggregation context and adding these context features to the query.

### Determination of Aggregation Context

The retrieval of learning resources is mainly based on query terms that have been explicitly specified by the author. However, with a growing amount of modules, which have already been aggregated, there are more and more implicit constraints that impact the suitability of other modules. For instance, if most existing modules are known to be specifically designed for schoolchildren, a module from adult education will less likely fit in. Many other metadata can be used similarly to judge the suitability of further modules, such as language, format or difficulty level. Some metadata fields, such as the difficulty level, are restricted to a specific vocabulary. Thus, these values are easy to use. Other fields, for instance the coverage field, are free text fields. Extracting semantics from free text entries from different authors is somewhere between tricky and nearly impossible. At least for fields like an age range, which is formally a free text field, a certain syntax might be implicitly followed.

As a method for formally determining the aggregation context, a selection $F_{cand}$ of candidate metadata fields of present modules is analyzed for frequent values. A threshold $T$ is applied for considering a repeatedly occurring metadata value as context constraint. That is, if for a metadata field a particular value $v$ occurs in at least a certain percentage of all aggregated modules, it is used as an context constraint. The threshold is suggested for tolerating false or missing metadata to a certain degree. Otherwise a single module with no or poor metadata would prevent that any aggregation context could be determined. Additionally, aggregation context should be utilized for retrieval only if some modules are already present in the aggregation.

A second implementation consideration is how context constraints are handled. They can be treated either as additional search terms or be weighted differently. The approach chosen here is to use a weighted formula. The query generates a score for each module; modules are ranked by a decreasing score.

$$score = \alpha * match(\text{query terms}) + (1 - \alpha) * match(\text{context}) \tag{4.1}$$

The weight $\alpha$ can be configured depending on how strong the influence of aggregation context should be in relation to original query components. A $\alpha$ value near 1.0 ensures that a change in the order occurs only if the original query matches similarly. It is assumed that original query terms are still more relevant than the context. An optimal value for $\alpha$ could be determined by a large scale experiment, by measuring the ranking performance of different values for a large number of queries.

The presented approach for utilizing aggregation context for query expansion is similar to the course-content situational relevance ranking metric by Ochoa et al. [OD07]. In contrast to Ochoa's metric, the method provided here does not take all metadata fields into account, but only a set of selected fields. These fields are considered to be consistent for a whole learning resource, such as language or the end user role. Other fields, such as learning resource identifier, creation time, or the typical learning time are supposed to differ too much. Ochoa uses metadata of existing course content for query ranking, which requires that the query processor is able to apply this measure. The approach of this thesis considers query expansion as a mechanism – query expansion can be used even if the query processor is not aware of aggregation context, because the client sends an expanded query to the processor. Thus, query expansion is suited to be used with existing repositories, which can be, for instance, accessed via SQI.

### 4.3.3 Improving Retrieval of Learning Resource by Estimating Adaptation Effort

Another way to improve retrieval of learning resources is to consider the costs that the usage of a particular learning resource causes. Learning resources are often required to be adapted in order to fit into an aggregation (details of adaptations are covered by Chapter 5). Each adaptation of a reused learning resource generates additional effort that has to be taken into account. The more adaptation effort a learning resource makes in a particular aggregation, the less useful it is assumed to be.

Consider an example of a user searching for a learning resource about business conventions in China. As the user wants to include the learning resource in a larger aggregation, the learning resource should follow his company's corporate design, be available in English and be intended for usage in on-the-job

training. What if the result of his query contains two learning resources, which both do not perfectly fit: one learning resource in German, and the other one designed for university teaching. Both learning resources cannot be applied unchanged; it is required to adapt them to the new context. But which one should be chosen?

This leads to a new metric for learning resource ranking. All known ranking metrics estimate how well a learning resource might be applied for a given task under the assumption that it remains unchanged. These ranking functions measure the usability of a fixed learning resource for a fixed task. The new metric instead estimates adaptation costs, or in other words, how much effort is needed for transforming a given learning resource to match the intended task. The assumption of this metric is that a given learning resource can possibly be adapted to suit a given task; and the costs of this transformation depends on which kind of adaptations have to be performed.

**Definition 4.1 (adaptation cost metric)**
*The adaptation costs metric ranks by increasing costs for transforming a learning resource to satisfactory match a given usage scenario. An optimal learning resource requires no adaptation costs at all.*

Foundation of this adaptation costs metric is Zimmermann's list of 15 relevant adaptation processes [ZBRS06]. The transformation of a learning resource into the desired state is performed by a sequence of some or all of the 15 adaptation processes. Adaptation costs are estimated by inferring the required adaptations from differences in particular metadata fields.

The need $n_a$ for performing an adaptation $a$ of module $m$ into the target state $t$ is calculated as the product of the metadata distance vector $\vec{d}_{m,t}$ between $m$ and $t$ and an involvement filter vector $f_a$ vector for adaptation $a$. The involvement filter vector contains values between 0 (metadata field has no impact on adaptation $a$) and 1 (a difference in this metadata field certainly requires the adaptation to be performed). The result is a scalar value that indicates if and to which degree this adaptation is required to be performed.

$$n_a(m,t) = \vec{d}_{m,t} * \vec{f_a} \tag{4.2}$$

The overall adaptation costs are obtained by summing up the individual costs for performing each single adaptation. The costs of an adaptation is a product of the need for that adaptation and a cost coefficient $b_a$. The cost coefficient $b_a$ depends on the available tools that are available to the user for performing the adaptation $a$.

$$c_{total}(m,t) = \sum_{a \in A} n_a(m,t) * b_a \tag{4.3}$$

Based on this cost estimate the learning resources may be ranked. Implementation details can be found in [ZMRS07].

## 4.4 Summary

This chapter has focused on the aggregation of modular learning resources. Aggregation is one of the six modular operations defined in Chapter 2. Aggregation is also the core of a particular authoring paradigm called authoring by aggregation. Existing approaches for authoring by aggregation have been analyzed. A comparison with the ADDIE model (a common model for the development of learning resources) has shown that existing authoring by aggregation processes support only one of the five development phases. One existing authoring by aggregation process has been extended to support further phases. The new process combines the known lightweight authoring by aggregation approach with a dedicated didactic design phase.

Beside the introduction of a didactic design phase, also the retrieval of reusable learning resources has been integrated more tightly into the authoring process. With this integration, additional information about the aggregation context becomes available for the retrieval system. Section 4.3 has developed a method that enables a retrieval system to prefer modules that are similar to those modules already present in the current aggregation. The assumed implicit context of an aggregation project (aggregation context) is inferred from the metadata of already present modules. This context information is utilized for query expansion. Another improvement of retrieval is proposed by taking the effort for adaptation into account.

# 5 Adaptation and Unification of Learning Resources

A third modular operation that has to be dealt with is adaptation. Adaptation means to transform a learning resource such that it suits a new learning or teaching context. Adaptation especially becomes relevant when learning resources from different origins are aggregated. In this case a mosaic effect is likely to occur: the result looks like a mosaic or patchwork; different designs, layouts, writing styles, target groups, etc. prevent aggregates to appear consistently. Consider again the scenario of setion 2.3.1: Sarah has retrieved several learning resources from different origins. She has aggregated these learning resources into one new learning resource. But until now, the aggregation has no consistent appearance. Sarah wants that her created learning resource uniformly conforms to the corporate design of her university. One solution to this challenge is to perform an adaptation of aggregated modules. While adaptation as such is not in the focus of this thesis, providing support for adaptation is.

Section 5.1 explains the background of adaptations. Requirements for a system that supports adaptation of learning resources are presented in Section 5.2. Section 5.3 presents a re-purposing framework, which allows to build re-purposing applications, such as an adaptation tool, more efficiently. Finally, the framework is compared to the requirements in Section 5.4. This chapter contributes to the overall goal of the thesis by improving interoperability and adaptation, as illustrated in Figure 5.1.

## 5.1 Adaptation of Learning Resources

Over the last years SCORM has become the de-facto standard for the exchange of learning resources. A core idea of the SCORM specification is to enable reuse of so called *Sharable Content Objects* (SCO). Ip et al. have though identified that reuse is practically prevented by the *mosaic effect*:

> "One problem holding back more widespread re-use of SCOs is the mosaic effect that arises when assembling a course using SCOs of different origins. SCOs (and any other type of re-

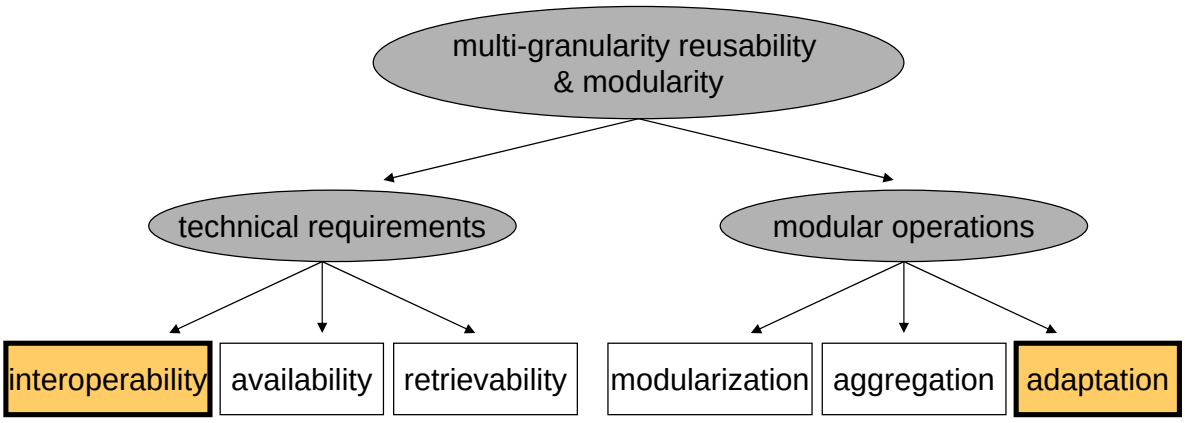


Figure 5.1: Contributions of Chapter 5 to multi-granularity reusability.

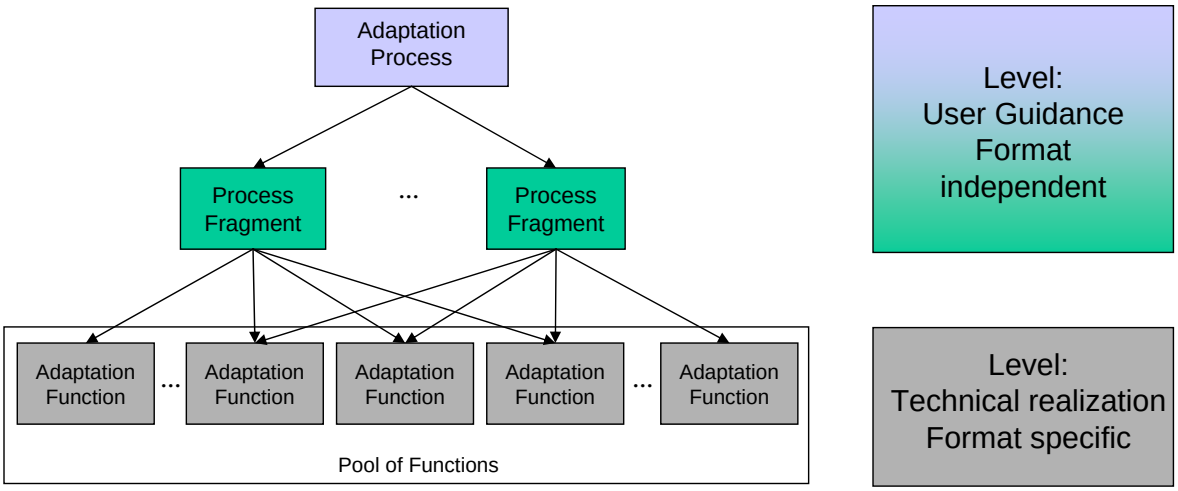


Figure 5.2: Structure of adaptation processes [ZBRS06].

usable learning object) are typically developed for a specific course and carry with them, a particular 'look and feel' that is consistent throughout the original course. The mosaic effect arises when a new course is constructed from a sequence of SCOs from originally different courses with different 'look and feel' characteristics. Hence, a 'mosaic' of different visual styles and interface elements is observed in the new course. The mosaic effect causes an unacceptably disjointed learning experience to be delivered to the student" [IRC03].

Whoever wants to reuse a SCO has to manually change its look and feel. The solution proposed by Ip et al. is to separate contents and styles by rendering the contents of a SCO via stylesheets. However, as SCO stylesheets are still not frequently used, this solution cannot be applied to the adaptation of existing learning resources. Furthermore, the mosaic effect covers much more than only the visual presentation of a learning resource. Also the pedagogical style, the language, terminology, or the degree of interaction may differ. Thus, it is often necessary to adapt a learning resource in more than just one dimension. Performing an adaptation of a module is one of the modular operations specified in Chapter 2.

Zimmermann et al. have analyzed which kinds of adaptations of learning resources are frequently needed and – even more important – how they are performed by users [ZBRS06]. For the target group regarded by Zimmermann there are 15 important adaptation types, which can be separated into two categories: structured adaptations that can be supported by completely or partially automated tools, and unstructured adaptations that are difficult to support by such tools. Structured adaptation processes can be internally organized in three levels: top level adaptation processes, process fragments, and adaptation functions. An adaptation process is constituted of several process fragments; and a process fragment is divided into a number of adaptation functions (figure 5.2). These adaptation functions are not dedicated to a single process or process fragment, but may be shared by several processes. Adaptation functions are e. g. identifying a graphical element, deleting a text fragment, changing a text or background color, or replacing one image by another one.

The analysis of adaptation functions has revealed that these functions can be described independent of the particular learning resource format and the adaptation process they are executed for. But nevertheless, adaptation functions have to be implemented over and over again for each new adaptation

tool. These properties of structured adaptation processes might be used to make the development of adaptation tools more efficient.

## 5.2 Requirements for Adaptation

The previous section has emphasized the need for adaptation of learning resources. There are some approaches, such as adaptive learning resources (e. g. the Multibook system [See02, ES01]), single source publishing or layout templates (e.g. Cascading Style Sheets) that facilitate the self-adaptation of learning resources to a few well-known scenarios. Adaptive learning systems are defined by Steinmetz as "learning programs capable of adapting themselves to the individual abilities of the learner, e. g., previous knowledge, interests, weaknesses or preferences with regard to forms of presentation" [SN04]. Most contents though are and probably will be available only in non-adaptive form. Thus, the adaptation of contents by authors is still frequently required.

A re-purposing tool is defined as a tool that supports an author in transforming a learning resource to suit a new learning or teaching context. A re-purposing tool guides the author to reach that goal; it should enable him to pursue the big picture instead of small editing steps. Developing such a re-purposing tool is a complex task. The tool has to work on different process granularities, from re-purposing over adaptation and modularization to modifications. A further challenge is the large number and the mix of different document formats. The most common format for Web based learning resources is SCORM (see Section 2.1.3). However, SCORM is only one format, and there are others, as well. Also, even if SCORM is used for specifying the structure of a learning resource, different formats, such as HTML, XML or Flash, have to be used for the actual contents. Developing tools for re-purposing is therefore a difficult and complex task.

Hence, it is not advisable to develop a new tool for each adaptation and each format combination completely from scratch. Instead, frequently used functionality should be moved into a framework, which enables the developer of a re-purposing tool to focus on the process level challenges of adaptations, instead on file and format handling details. Instead of implementing yet another monolithic tool, the tasks of a re-purposing tool are divided into two parts: a basic re-purposing framework is responsible for the low-level functionality, whereas a re-purposing application on top of that framework can focus on high-level tasks.

This section analyzes the requirements on a common foundation for re-purposing tools. A concept for an abstraction model for learning resource contents and modifications of that contents is developed and presented.

### 5.2.1 Requirements for Abstraction of Content Representation and Adaptation

An abstraction model is intended to simplify the development of re-purposing applications. Especially adaptation tools (see Section 5.1) and modularization tools (Chapter 3) are considered as re-purposing applications. This subsection analyzes the requirements that are posed to a re-purposing framework that is intended as foundation of re-purposing applications.

The main goal of a re-purposing framework is abstraction. Abstraction helps developers of re-purposing applications to reduce the complexity of these applications by outsourcing some tasks to the framework. Besides the facilitation of application development, it is also required to avoid that re-purposing applications are limited to a single document format; bridging the gap between the application and a particular format implementation is again an issue of abstraction. In addition, many Web based learning resources, such as SCORM packages, consist of several interconnected documents. The user of a re-purposing tool, though, does not want to process one file after the other, performing the same task multiple times in sequence. Instead, he regards a learning resource – even if consisting of multiple files – as a single resource and also desires to process it in one step. Overcoming document boundaries is yet another abstraction that is demanded of the re-purposing framework. But in the end, the re-purposing framework should have no side effects. In particular, the usage of the framework should not lead to an unintended loss of information. When abstraction is concerned, systems sometimes tend to ignore and discard detailed information that is not part of the abstract model; however, a loss of this information could invalidate a learning resource.

Thusm, the core aspects of the requirements can be subsumed in five items:

- Format-independent abstraction model
- Support for structured multi-document resources
- Enhanced support of content analysis
- Modification of content without unintended information loss
- Extensibility

These five requirement items are now explained in more detail.

### Format Independent Abstraction Model

The intended abstraction model must be format-independent in a way that an application using the framework will be enabled to deal with resources in different formats. Ideally, the application should not require any knowledge about the particular document format of a learning resource.

This means that a format abstraction is needed that covers all relevant formats. For each supported format a mapping from the format-specific data to the abstract model must be given. However, an application must still be able to analyze and modify all details of the underlying format. Therefore the abstract model should be extensible or shapeable for particular formats.

Even though the framework supports format-abstraction and extensibility, there is still a demand for determining the type of supported resources. The focus lays mainly on text-based learning resources, which means everything that contains a major share of written text. Audio and video elements are only considered as embedded elements in text documents. Manipulating audio and video data is very different from modifications of text-based documents. Thus, the manipulation of audio and video cannot be covered by this thesis.

### Support for Structured Multi-Document Resources

As stated in the previous section, a learning resource may consist of multiple documents. Usually, there are links between these documents, which turn a set of single documents into a meaningful ensemble. Those links may be hyperlink-like references between content-documents or references in a manifest document. One example for multi document learning resources are SCORM packages. A SCORM package contains a manifest file (an XML document), which refers to e. g. several HTML documents. Each HTML document may include media objects, such as images or flash animations, and may link to other HTML documents. Another example are completely HTML-based learning resources. In this case, the logical structure of the learning resource is typically represented by menus on each single HTML page, providing links to other HTML documents.

The goal of the intended architecture is to facilitate content analysis, adaptation and modularization. Therefore it is necessary to regard several documents as one logical resource as well as to give the possibility to regard each single document. The application should be enabled to process the whole learning resource as a logical resource; that is, structure, contents and relations between documents should be observable. This logical view has to span the different files and formats.

### Enhanced Support for Content Analysis

An intelligent re-purposing tool, which supports the user in reaching his goals, has to obtain some information about the learning resource it deals with. It would, for example, be helpful for restructuring tasks if the structure of a course could be shown to the user; possibly enriched with information about the relations between different text blocks (e.g. coherence of text blocks or how they are pedagogically related to each other).

Consequently, support for content analysis is one of the most important features of a re-purposing framework. Available format libraries allow navigation through a document's structure. But navigation alone is not sufficient. A query mechanism that enables the application to ask for more complex information is also needed.

Two concurrent analysis approaches are proposed: The first approach is that an application queries for information. In this case, the application has to know what to ask and how to interpret the result. As a second approach it should be possible to perform analysis methods within the framework, independent of the application that uses the analysis results. Using this approach, the application can query for inferred knowledge; it does not need to care about how that knowledge has been created.

The whole learning resource, consisting of several documents, can be regarded as a single logical content. Therefore, a query may combine data from multiple documents. Queries should work on abstract structures and properties instead of format-specific data bindings.

### Modification of Content Without Unintended Information Loss

Analysis of the content is only the first step of re-purposing. The data has also to be changed by the application. These changes should take place without losing information. This is a challenge because the analysis operates on an abstract data model.

If changes were performed on the abstract data model, this abstract data would have to be converted afterwards back into the original format. Such a transformation would cause a loss of information in most cases, because an abstract data model would not represent all details of all arbitrary formats. Thus, changes cannot be performed on the abstract data model. Instead, a way has to be found to bridge the gap between the abstract view on the contents by the re-purposing application and the format-specific details of the actual physical contents in the documents.

### Extensibility

The re-purposing framework should be an open framework for various applications. Hence, extensibility should be considered as one of the core requirements. Extensibility comprises especially

- Extensible towards new document formats
- Applicable towards new re-purposing processes
- Extensible towards new content analysis methods

First of all, support for new particular formats should be addable without changing the applications on top. Existing applications should then automatically be able to use the new formats. Second, the framework must be flexible enough, so that new applications for different purposes can be implemented on top of the available framework. And finally, there should be a mechanism to integrate new analysis methods as framework components. These components may change the abstract data model in order to provide more and better information to the application.

## 5.2.2 Related Work

Abstraction is a common principle for dealing with complex challenges. Introducing layers of abstraction can be used to reduce complexity. Steinmetz identifies the goal of abstraction of multimedia data as an integrated and unified way of description and treatment of all media [Ste00]. For the intended approach – an abstraction model for learning resource contents and for modifications of these contents – two kinds of abstraction are of interest: static abstraction of document contents, and dynamic abstraction of data manipulation. For the manipulation of a document, a logical representation of that document is best suited, as Brugger has mentioned:

> "Ein Dokument lässt sich am leichtesten auf der logischen Ebene manipulieren. Die interessantesten Anwendungen (...) benötigen die logische Struktur." [Bru98]

There are some representation formats for logical documents, such as the Standard Generalized Markup Language (SGML) [Gol90] and Open Document Architecture (ODA) [FFK92]. SGML describes

only the logical structure of a document, it does not specify how the contents are transformed into a physical document for presentation. In contrast, ODA specifies not only the logical structure, but also the physical structure of a document. Today, XML[17], which is a subset of SGML, has become a popular logical document format.

SGML and XML are simple but powerful models for specifying logical documents. The resulting documents are human-readable and can be automatically validated if a proper document specification exists. It is even possible to combine several specifications – this would facilitate the representation of multiple documents as a single resource and the attachment of annotations. There are also suitable query mechanisms for XML (e. g. XPath[18] and XQuery[19]). A drawback of XML is that it is strictly tree-oriented. If non-tree graphs have to be represented the graphs have to be transformed into a tree, which makes processing and querying more difficult. Non-tree graphs occur, for instance, if semantic relations between elements are inserted into a tree. The same problem applies to ODA. A versatile model for specifying and processing graph-like information is the Resource Description Framework (RDF) [MM04]. RDF graphs consist of a set of statements. Each statement is a triple that connects two nodes of a graph by a relation. There are also schema specifications for RDF; for more sophisticated knowledge representations, the Web Ontology Language (OWL)[20] can be combined with RDF. OWL can be used to model ontologies, and to utilize them for reasoning.

Although physical and logical documents can clearly be separated in theory, it is not always that simple in practice. For instance, HTML is another famous SGML derivate and thus also a logical document model. However, HTML mixes logical structure with presentation and layout information. Some further formats, such as the PDF, are more oriented towards layout than towards logical structure. There are methods for discovering the logical structure of physical documents [Sum98, Bru98]. These methods can also be applied to documents that combine physical (layout-orientation) and logical properties.

There are approaches for the modeling of dynamic behavior, which can serve as basis for the abstraction of content modifications. The Model Driven Architecture[21] approach separates application logic from underlying platform technology. Platform independent models document the behavior of an application separated from the technology-specific implementation. Model transformation in the sense of the Model Driven Architecture approach can be seen as an application of a graph transformation [RN06]. This approach has found a large community and is used in different applications. Beside the different scenario, the idea to abstract from the implementation and specify generic models is also the motivation behind the approach presented here. It abstracts from format-specific resources and format-specific modifications of these resources by generating a resource model and by modeling the modifications which can be performed on the resources.

The ALOCoM framework [VGJD05] is an ontology based framework to enable reuse of learning objects. The framework is focused on slide presentations. A slide presentation is disaggregated into its components and mapped to an object model. Components are reused by copying these components into a new slide presentation. This scenario allows the reuse of complete slide presentations, and of parts of these slide presentations, e. g. a single image or text block. The generated slide presentation uses

[17] http://www.w3.org/TR/2006/REC-xml-20060816/
[18] http://www.w3.org/TR/xpath20/
[19] http://www.w3.org/TR/xquery/
[20] OWL Web Ontology Language, http://www.w3.org/TR/owl-features/
[21] http://www.omg.org/mda/

default presentation styles. The major difference between the content representation proposed in this section and ALOCoM is that ALOCoM does not support the modification of learning resources beyond the functionality provided by existing tools (e. g. Microsoft PowerPoint).

Kashyap and Shklar [KS01] propose an RDF model based approach to adapt content resources for different devices. In their work they use a representation of the features of the different devices and components which represent the content resources. A XML resource can be adapted to the different devices using device-specific style sheets. Depending on the device which is requesting content resources an appropriate style sheet can be generated based on the information in the RDF model. No library or collection is needed, containing specific style sheets for all the possible requirements a device might have. This approach follows the idea of uncoupling information from presentation and to adapt certain properties of a content resource; it focuses on Web applications.

## 5.3 A Framework for Abstraction of Learning Resource Content Representation and Adaptation

A learning resource abstraction model has been specified that satisfies the requirements of Subsection 5.2. The model consists of two parts: a static content representation model and a dynamic modification model. The content representation model specifies how a learning resource, possibly consisting of multiple documents, is transformed into an abstract representation of logical structure and contents of the document; this model allows also to add relations and other annotations about the contents to the content representation. The modification model defines how the contents can be modified by re-purposing applications based on knowledge about the abstract content representation.

The two parts of the learning resource abstraction model are described in detail below. First the abstract content representation is explained. Afterwards, the modification model is presented based on a preceding analysis of modification granularity.

### 5.3.1 Abstract Content Representation

As specified in the requirements, the abstract content representation should be applicable to different text-based document formats. Applications that builds upon the abstract content representation model should not necessarily require information about the particular format of the learning resources they process. The extensibility requirement demands that the set of supported document formats can be extended without much effort. The content representation should comprise all documents a learning resource consists of within one common representation. In addition, annotations that have been generated by applications or internal components of the framework should be part of the content representation and be available for content queries.

The content representation has to contain all elements of a learning resource that a re-purposing application is interested in. The level of detail should be controllable by the application. Because of performance reason, it is advisable to reduce the level of detail for some tasks. The content representation should at least comprise the logical structure of the documents, textual contents, properties of media objects, and presentational information (e. g. fonts, colors, layout). Thus the level of detail of the content representation may reach down to single text fragments and embedded media elements.

Re-purposing applications have to know some details about the elements of the content representation. For instance, it is necessary to know which elements are small text fragments, media elements, combined blocks, or even whole sections or chapters. A text fragment can be formatted in different ways, and also the meaning may differ: a headline might be treated different than normal text or an image caption. These demands and the need for format abstraction require that elements are classified by abstract element types. The specification of types and how they relate to each other is called Content Ontology (CO). There are certainly different domains of types, which may be if interest, for instance structural types, presentational types, pedagogical types, and even a topical classification. Besides the types, also the relations between types and instances of these types have to be expressible. A particular Content Ontology for abstract content representations has been developed by Bergstraesser et al.; more details about the development of ontologies and the resulting Content Ontology can be found in [BFRS06].

An abstract content representation, which contains relations between its elements is essentially a graph. Therefore, graph models, such as RDF are better suited for this task than tree models, e. g. XML or SGML. For realizing a re-purposing framework a static content representation model is not sufficient; efficient mechanisms for creating a content representation out of the original learning resource and performing modifications of the contents according to the known requirements have to be considered. Out of this reason, the content representation has been decided to be established on three representation layers with increasing level of abstraction. The proposed layers are (see Figure 5.3):

- A physical representation layer which represents the involved files as they exist on disk
- An object-oriented representation layer that represents the content structure as an object tree in a format-related manner
- A semantic representation layer that represents the abstract logical document structure and contents as a semantic graph enriched by known relations and annotations

On the object-oriented representation layer a so-called Object-Oriented Content Representation (OOCR) will represent the structural entities of the documents as a tree of objects. These objects are also the entities which provide methods for modifying the content.

The semantic representation layer consists of a graph-like representation – for instance an RDF model – called Semantic Content Representation (SCR) that contains the structural information of the OOCR plus additional semantic markup and relations. The SCR also provides the required format abstraction by using a generic Content Ontology. The Content Ontology will provide the necessary concepts for interpretation of the SCR by re-purposing applications.

However, although an RDF model is very helpful for analysis, it is not suitable for changing the content. If changes were performed directly on the Semantic Content Representation, it would be necessary to transform this model back into the original format – which would lead to information loss. Therefore, all modifications have to be propagated to the OOCR, where they can be performed format-specific. How this propagation can be realized will be described in more detail below.

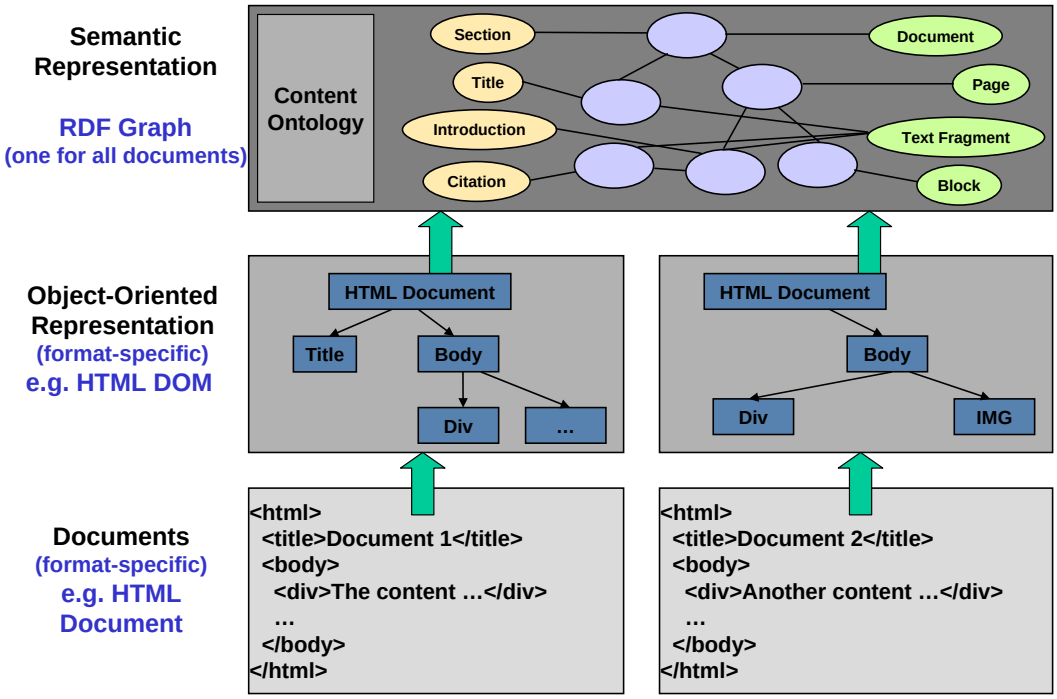


Figure 5.3: The three layers of the learning resource content representation.

### 5.3.2 Granularity of Modifications

An important design decision is the granularity of modifications. Is, for example, the replacement of a corporate design a single modification or a combination of several modifications? Zimmermann et al. have identified a structure of adaptation processes, which is helpful for the consideration of granularity [ZRS06]. The structure is illustrated in Figure 5.2. On the most general layer, whole adaptation processes are resident, e.g. the adaptation to a different corporate design. An adaptation process divides into several process fragments. Process fragments are composed of adaptation functions, which may either read or modify the contents of a learning resource. Which of these granularity levels is best suited for modeling of content modifications?

The goal of modification modeling is to provide an abstraction layer for separating the concerns of re-purposing tools and the format-specific content modification methods. Also, reuse of modifications, which are implemented once and reused for several re-purposing applications, is a central motivation. In this respect, adaptation processes are too large to be reused easily and often. Process fragments are also application dependent, may rely on information from other process fragments, and sometimes comprise interaction with a user. They are also reused rarely. Adaptation functions, finally, are reusable for multiple process fragments, do not necessarily require user interaction and need only a manageable amount of parameters to work. Therefore, content modifications are best modeled at the granularity of adaptation functions – restricted to those adaptation functions which cause changes of the content. These modifications are mainly insertion, deletion, replacement and rearrangement of elements, as well as changes of attributes and relations.

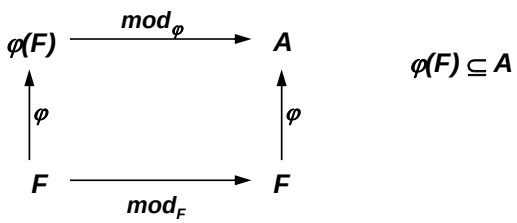


Figure 5.4: Transformation into abstract content representation space.

### 5.3.3 Theoretical Model of Modifications

The Learning Resource Content Representation is on the top layer a graph and is a transformation of the whole learning resource contents, containing all information, which is relevant for performing adaptations. It is assumed that modifications at the granularity of adaptation functions produce only delimited local changes of the Learning Resource Content Representation. These changes can be expressed as graph operations. Consider there is a learning resource $r$ in which one logo should be replaced by another one. If the learning resource consists of HTML documents, a logo is usually embedded by using a reference to the image file, which contains the logo. Replacing an image in HTML documents requires only changing the image reference. For other formats (e.g. Microsoft Word), images are physically embedded in documents; hence a replacement works different. Let $H$ be the set of all valid HTML documents and $W$ the set of all valid Microsoft Word documents.

$$exchangeLogo_H : r_1 \mapsto r_1' \quad |r_1, r_1' \in H$$

$$exchangeLogo_W : r_2 \mapsto r_2' \quad |r_2, r_2' \in W$$

And for the general case:

$$mod_F : r \mapsto r' \quad |r, r' \in F, mod \in M$$

where $r$ is a document from a given format space $F$ and $mod$ is a modification out of the set of all modifications $M$.

Consider the projection of learning resource $r$ into the Learning Resource Content Representation (LRCR) $r \mapsto \varphi(r)$, where $\varphi$ is the projection function from the document format space $F$ into the abstract LRCR space $A$. The modification from the previous example can now be observed in the LRCR space. The function, which modifies $\varphi(r)$ into $\varphi(r')$, is called $mod_\varphi$ and represents an abstract modification of the learning resource content.

This algebra helps developing content adaptation tools. Adaptations have no longer to be implemented directly as format-specific methods. Instead, an adaptation tool analyzes $\varphi(r)$ (the LRCR) and specifies adaptations as a concatenation of modifications $mod_\varphi$. Each modification mod is transformed by an underlying layer into a format-specific modification $mod_F$. This transformation from LRCR space into the actual document format space is also called interpretation of an abstract modification. Fig. 5.5 illustrates these transformations.

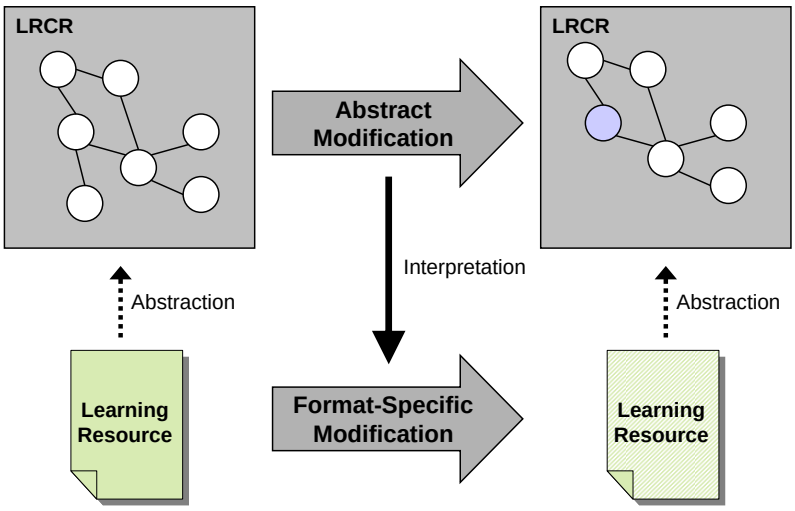


Figure 5.5: Interpretation of abstract modifications.

## 5.4 Discussion

This chapter has introduced a concept for an abstract learning resource content representation and for modifications of the content. In the beginning, five requirements for the content representation have been defined. The presented concept has still to be checked if it fulfills these requirements. The requirements, as defined above, are format-independence, support for multi-document learning resources, enhanced content analysis support, content modifications without information loss, and extensibility. The realization of each of the requirements will be separately analyzed below.

- **Format Independent Abstraction Model.** The top layer of the content representation is independent of the actual document formats of a learning resource. The same can be stated about the way modifications of the content are performed by re-purposing applications. For a re-purposing application, the document formats are not relevant; only the lower layers of the framework have to deal with format specific issues. Thus, the format independence requirement is met by the concept.

- **Support for Structured Multi-Document Resources.** If a learning resource consists of multiple documents, the lower two layers of the content representation keep the contents of each document separate from the other documents. But the SCR finally joins all contents into a single representation. The elements of multiple documents get connected by relations according to their logical or semantic relationship. Re-purposing applications may handle a multi-document learning resource as a single, coherent resource.

- **Enhanced Support for Content Analysis.** Support for content analysis is achieved by two means. First, the Semantic Content Representation in form of an RDF graph enables re-purposing applications to query the contents. There are several query languages for RDF that can be used to extract information about the elements of a learning resource. And second, it is possible to enrich the SCR by additional framework components before the application accesses the SCR. These components specialize on particular analysis methods and can be reused by different re-purposing applications.

- **Modification of Content Without Unintended Information Loss.** Modifications of contents are specified by a re-purposing application as an operation on the SCR graph. The specified modification is then propagated to the OOCR layer, which interprets the modification in a format specific way. This format specific execution of modifications ensures that no information is unintentionally lost.

- **Extensibility.** Extensibility is supported by the content representation concept through different mechanisms. The framework enables to support additional document formats easily. It is only necessary to provide a transformation method for mapping a document into the SCR and format specific implementations of modifications. Further re-purposing applications are also supported; a new application simply has to analyze the SCR and perform modifications of the contents. It is even possible to specify new kinds of modifications if some are missing for a new application. New content analysis methods can be integrated, because each analysis method uses the existing content representation as input and writes the results into the SCR.

The overall result of this comparison is that the proposed framework fulfills all demanded requirements in theory. Chapter 7 will present an implementation of the abstraction framework and of an adaptation tool on top of the framework. This implementation has to finally demonstrate if the approach is feasible.

# Part III

# Improving Retrievability by Metadata Generation

# 6 Metadata Generation

Before a learning resource can be reused by any user, it has to be found within a repository. The larger repositories grow, the more important become retrieval methods for learning resources. Retrieval generally means to match a user query against a set of objects (e. g. learning resources). Retrieval systems for general Web resources, for instance Google[22], use mainly a full text index for matching a query, plus some additional information for ranking the result entries [BP98]. In the case of learning resources, many facts beyond the occurring words in the text are of relevance. Firstly, a topical categorization of a learning resource can be more relevant than the occurrences of words. Secondly, pedagogical metadata could be useful as part of user queries. Thus, metadata is even more important in learning resource retrieval than in Web retrieval. Furthermore, learning resources do not always consist mainly of text. However, this thesis focuses on learning resources with a high share of textual content.

Unfortunately, for many learning resources only few metadata fields are available as studies have revealed [NTD03, SGP+05]. In order to improve the retrieval quality, more metadata has to be provided. There are basically two ways to create this metadata: manual labor or automation. Metadata records can be completed by humans; or automatic metadata generation methods are employed to extract or generate additional information. We believe that automatic metadata generation can reduce the costs of metadata creation and makes reuse more beneficial. Liddy et al. have shown that automatically created metadata may in some cases be qualitatively comparable to manually created metadata [LCF+05].

This chapter proposes a new metadata generation method for learning resources, which is based on a free encyclopedia (e. g. Wikipedia) as substitute corpus. The method generates topical metadata – more precisely a topical categorization of the learning resource content. In the first section some related work about methods for information retrieval and machine learning in general, and particularly metadata generation for learning resources is discussed. Afterwards, Section 6.2 introduces the concept for a new metadata generation approach. The novelty of this approach is that a traditional training corpus for

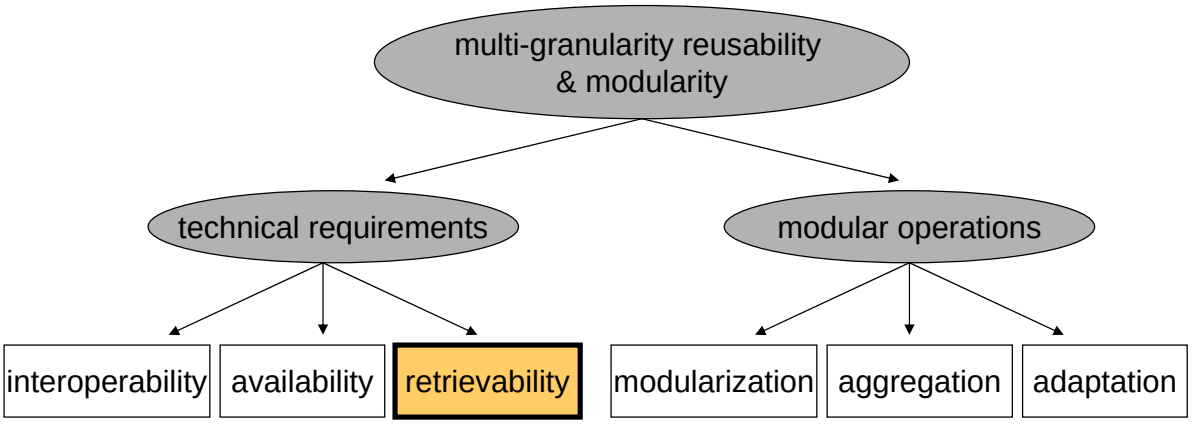


Figure 6.1: Contributions of Chapter 6 to multi-granularity reusability.

[22] The Google search engine: http://www.google.com

machine learning methods is replaced by Wikipedia articles as substitute corpus. The categorization method is calibrated by determining parameter values that result in the best categorization performance in Section 6.3. The method and the determined parameters are evaluated in Section 6.4. Section 6.5 covers the generation of pedagogical metadata. Finally, the chapter is concluded by summarizing the contents and giving an outlook. This chapter contributes to the overall goal of the thesis by improving retrievability of learning resources, as illustrated in Figure 6.1.

## 6.1 Related Work on Information Retrieval, Machine Learning and Metadata Generation

Automatic generation of metadata is strongly related to methods in the areas of information retrieval and machine learning. Methods and systems that allow users to search for resources are subsumed by information retrieval. Information retrieval systems typically rely on precalculated indices or metadata for query processing. Machine learning comprises methods that learn to perform particular tasks from given examples; generation of metadata is such a task. One of the most important purposes of learning resource metadata is to improve the retrieval of learning resources. A metadata record contains structured information about the corresponding learning resource. By querying specifically for particular metadata fields, a query can be more precise than a simple keyword based full text search over the whole contents of a learning resource. Therefore, some background knowledge about these methods is provided first, before their application for automatic metadata generation is explained.

### 6.1.1 Information Retrieval

According to Baeza-Yates and Ribeiro-Neto *information retrieval* "deals with the representation, storage, organization of, and access to information items" [BYRN99]. In the scenario of this thesis, information items are learning resources. An information retrieval system supports a user in finding those information items, which he is interested in. In order to perform the retrieval task, the user's needs or interests have to expressed as a query, which can be processed by a retrieval system. Baeza-Yates and Ribeiro-Neto divide retrieval tasks into information retrieval and browsing [BYRN99]. Information retrieval refers to searching for information by formulating a query and obtaining a result list. Browsing means to follow relations or links between objects of interest.

The information that an information retrieval system has about a resource for matching a query is called *logical view* by Baeza-Yates and Ribeiro-Neto. The logical view can, for example, be the full text of a document, only a set of keywords, but also further attributes. The specification of the logical view for a retrieval application influences which kind of queries are allowed and how they are processed. Retrieval interfaces of learning object repositories often exclusively use the metadata records of stored learning resources as logical view; but full text retrieval is also sometimes supported. For full text search, all words that occur in a document are potential index terms for that document; a document can be found by searching for one or multiple of these index terms. But even for full text retrieval it is reasonable to reduce the set of index terms. This is usually achieved by text transformation methods, such as stopword elimination (very frequent words like "the" or "a" are ignored) or stemming (different derived forms of

a word are reduced to their root form). However, full text retrieval systems may become quite complex [BP98].

Sometimes, the set of index terms has to be even more reduced. A reduction of index terms decreases the memory demands of an index and also the search time. This can be achieved for instance by determining keywords – the most relevant terms that describe the contents – and use only these for indexing. Keyword extraction methods are widely known [Tur00]. In some cases, the relevant index terms are not words that occur in the text, but different attributes, such as a document type, genre, file size, creation date, or the language a resource is written in. Several approaches are known that generate this kind of attributes, for instance [SFK00]. Many of these approaches employ machine learning methods, which are explained below.

A special branch of information retrieval is multimedia information retrieval. In multimedia information retrieval, multimedia resources are the information items of interest. These objects have different attributes than text-based resources. Also, different methods for attribute extraction and query matching are required [Sch05]. Retrieval of learning resources can benefit from multimedia information retrieval insofar that pedagogical attributes also require a more complex treatment than attributes in simple text retrieval.

The Learning Object Metadata (LOM) specification provides a standardized format to describe learning resources. LOM records are often generated by authors and delivered together with the learning resource. Therefore, metadata records are predestinated as logical view for learning resource retrieval. Many repositories rely only on metadata records for retrieval; particularly the availability of the LOM standard improves retrieval for users and developers. The drawback of using only metadata for retrieval is that a lack of metadata can make a learning resource undiscoverable. The generation of additional metadata is therefore an important task in order to ensure retrievability of learning resources [DH03].

### 6.1.2 Machine Learning

Information retrieval often relies on precalculated metadata about resources. Metadata can be automatically generated by different types of methods. One popular technology that is frequently applied for this purpose is machine learning. Machine learning basically comprises all methods where a systems learns how to perform a tasks by experience. According to Mitchell a system is considered to learn from experience, if its performance at given tasks improves with experience [Mit97]. Within the large field of machine learning methods and applications, one particular task is *concept learning*. Concept learning means learning a general mechanism for detecting a concept (or category) based on given samples of members and non-members. A common concept learning task nowadays is spam filtering: a spam filter takes a set of e-mails as input, which are manually labeled as spam or non-spam e-mails [SDHH98]. The spam filter learns from this input how to classify new e-mails as either spam or non-spam; spam e-mails are either marked as such or immediately deleted. In this example, a machine learning system tries to learn the mechanism *'detect spam'*.

Sebastiani has summarized known machine learning based methods for automated categorization of textual resources [Seb02, Seb05]. Text categorization is the task of assigning a given text document to one or multiple predefined categories. If the categories are not predefined and need also to be automati-

cally determined by an algorithm, this task is called text clustering. For the rest of this chapter we stick to Sebastiani's definition of text categorization, which requires a predefined set of categories[23]. Sebastiani further assumes that categories are simply labels, which have no further meaning. Sometimes, categories may also be hierarchically organized; for instance, Web pages are often classified in hierarchical categories [DC00]. Text categorization methods can be divided into single-label approaches and multi-label approaches. Single-label approaches assign each document to exactly one category; with multi-label approaches a document may belong to multiple categories. Thus, multi-label methods allow overlapping categories. A special case if single-label categorization is a binary classifier; it decides if a document belongs to a particular category or not[24]. Another categorization type is ranking categorization: instead of finally assigning categories to a document the categories are only ranked by their estimated relevance. The finally decision on which categories are suitable is, for instance, left to a human expert.

Machine learning methods for text categorization perform the categorization task by automatically building a suitable classifier based on a set of training documents. These training documents are sample documents, of which the category assignment is already known (e. g. because they have been manually labeled). For a qualitative evaluation of classification methods the effectiveness has to be measured. Typically, a corpus of labeled documents is divided into two or three sets. In the simple case, the corpus is divided into a training set and a test set. The training set is used to learn the classifier. Afterwards, the learnt classifier is applied to the test set for measuring the classification performance. Sometimes, there are also additional parameters of the classification method, which have to be optimized; in this case, the corpus is split into a training set, a test set and a validation set. First, the parameters are tuned using the training set and validated with the validation set. Again, the test set is employed to evaluate the performance of the learnt classifier.

The effectiveness of a classifier is typically evaluated using measures such as precision, recall, and combinations of them. Precision and recall are formally defined as probabilities that a randomly chosen document $d_x$ falls into particular categories [Seb02]. Precision is defined as the probability that if a randomly selected document $d_x$ is categorized into category $c_i$, this decision is correct. Recall on the other hand measures the probability that a randomly selected document $d_x$, which belongs to category $c_i$ is categorized into $c_i$ by the classifier. One of the two values can always be maximized at the expense of the other. Hence, there are further measures that combine precision and recall to one joint measure. For instance, the $F_1$ measure calculates the harmonic mean of precision and recall. Furthermore, to evaluate the overall performance of a classifier over all categories, the precision and recall values for the individual categories have to be aggregated.

What has been said above for information retrieval is valid also for machine learning: documents are not processed as continuous text, but are transformed into another representation, which can be better processed. In the case of text categorization, a document is normally represented as a vector of weighted terms. According to Sebastiani the main difference between different approaches is what is considered as a term and how to compute the weights [Seb02]. In the more general case, machine learning methods expect tasks as input; a task is often represented as a vector of features, where a feature can be any attribute that can be calculated for a given input task.

[23] This classification type is also called supervised learning, because the output of a classifier is compared to known values. Unsupervised learning refers to clustering, where the resulting categories are not a priori known.

[24] From a theoretical point of view, every multi-label classification task can be mapped to a number of binary classifiers.

When machine learning technologies are intended to be applied, several considerations have to be made: Which training corpus is used for building a classifier? Which categorization method is applied? Which features are used for representing documents? How will the classifier performance be evaluated? And finally, some categorization methods have to be tuned with additional parameters. Different categorization methods are known. Common methods are, for instance, probabilistic classifiers (e. g. Naïve Bayes), decision tree classifiers (e. g. C4.5), support vector machines, neural networks (e. g. perceptron), or example based classifiers (such as k-nearest-neighbors). Details about these methods can be found in [Mit97] and [Seb02]. Some sophisticated categorization approaches use structural information documents [Kru01b], or natural language processing methods [GM07b] for classifying documents.

### 6.1.3 Approaches for Automatic Metadata Generation

After information retrieval and machine learning have been described in general, this subsection gives an overview how these approaches are used particularly for the generation of metadata. Metadata generation is a field of research that has been heavily worked on in recent years. There are many approaches for metadata generation for documents in general [Nou05] and for learning resources in particular [Ber05]. Metadata generation methods can be classified by the type of metadata to generate, by the sources that are used, by the required prerequisites and the applied methods.

Possible target metadata types are, for example, content-related metadata (such as title, keywords and categories), process-related metadata (author, creation date, version) or – in the application domain of e-learning – pedagogical metadata (learning objective, target group, difficulty, activity level). Sources for metadata generation strongly depend on the target metadata types. Content-related metadata requires to analyze the contents of a document, whereas process metadata, such as author and creation date can be obtained from the authoring environment [HHRS05].

Metadata generation methods are applied in many different application areas, for instance in multimedia databases or digital libraries. The state of the art in metadata generation applications for digital libraries has recently been reviewed by Greenberg et al. [GSC06]. They distinguish between metadata extraction and metadata harvesting. *Metadata extraction* subsumes all methods that mine metadata out of the content of a resource. If metadata is already existing in some form, the process of collecting this information is called *metadata harvesting*. Library documents often meet particular document structures – for instance the title and author of a journal article typically appears on top of the first page, whereas references to other literature are located at the end. The document structure can be exploited if the genre of a document is known.

Content-related metadata is the most important type of metadata for retrieval of documents and especially learning resources. Users often search by words that describe the desired subject of learning resource. Common content-related metadata fields are title, keywords, classification and an abstract or brief description. Using these fields can be more efficient than using full text search, because full text search may offer resources where an entered search term is only slightly related to the overall topic of the resource. The discussion of content-related methods will focus on keywords and classification. Keywords are terms that give a hint on the topics that are covered by a document; these keywords can be any words without restrictions. Classification, in contrast, is restricted to a fixed set of classes, from

which concepts can be taken to describe the contents of a document. Hence, the methods for generation of keywords and classification information also differ: for classification a mapping to known terms is required, whereas arbitrary words may be produced as keywords. Classification metadata can, for example, be generated automatically using machine learning classifiers. Another approach for the classification of documents are rule-based systems, such as the ontology-based metadata generation described in [SvH01]. If metadata is intended to be understood by humans, categories should have a meaning and name that is obvious to humans. Ontologies and taxonomies may serve as structured sets of meaningful concepts. Faatz describes how domain ontologies are initially generated and how they can be extended at a later point in time [Faa04].

The unsupervised equivalent to classification is clustering. Clustering algorithms calculate a distance between documents and build groups of documents, which are near to each other or have common attributes. A common clustering method is the K-means algorithm. A method for clustering news articles is presented in [NCSS06], where a set of 400 clusters is calculated based on the co-occurrence of entities, such as persons, organizations and places. Each of these clusters represents a hot topic that has been extensively discussed in the media.

Keyword extraction methods can be classified by their coverage: Domain-dependent methods are limited to a particular knowledge domain but usually provide better results. Domain-independent keyword extraction methods can be applied universally, but are less precise. Domain-dependent methods are based on a domain model, which contains relevant terms and further information for the particular domain. Documents are searched for these terms for determining keywords. [Kru01b] demonstrates how to build a domain model out of existing documents. Matsuo and Ishizuka have introduced a domain-independent method for extracting keywords from a single document without having a large corpus of documents [MI04]. This approach is based on the specific distributional characteristic of terms. Sometimes, the term keyphrase is used as a synonym for keyword. Turney defines "*automatic keyphrase extraction* as the automatic selection of important, topical phrases from within the body of a document." He considers keyphrase extraction as a special case of keyphrase generation [Tur00]. Keyphrase extraction produces keyphrases, which necessarily have to appear in the examined document. Another technology for extracting keywords from Web pages is to exploit the structure of a document [Kru01a]. Opposing to the approaches above, Kruschwitz uses only those terms as keywords, which appear in at least two different contexts within a document; the considered contexts are meta information, document headings, document title and emphasized parts of a document.

Automatic metadata generation for e-learning applications mostly relies on generic approaches for metadata generation, such as those described above. In some cases, knowledge about genre specific properties of learning resources is used to tune these metadata generation approaches.

Saini et al. , for instance, employ a Naïve Bayes classifier to categorize learning objects [SRS06]. They use two existing topical taxonomies with a total of 41 categories and 412 learning resources that have been collected on the Internet. The Naive Bayes classifier achieves a performance (F1 measure) of about 43% for the overall learning task. A clustering method increases this measure to 60%. Khoury et al. use natural language processing methods for extracting metadata from learning resources [KKK06]. Their approach is based on a part-of-speech analysis of sentences. The extracted information is finally used to predict the title, description and keywords for a document.

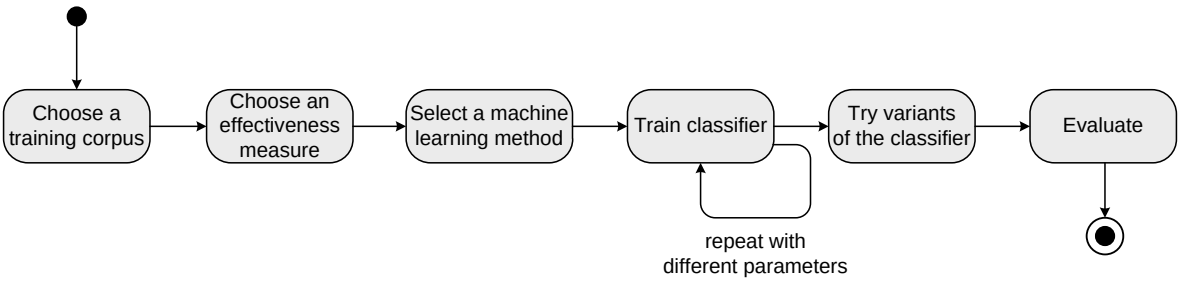


Figure 6.2: Process of training and evaluation of a classifier.

Another work has to be mentioned: the Simple Automatic Metadata Generation Interface (SAmgI) [MDO07]. SAmgI is not a single metadata generation method, but a framework for metadata generation methods. Different automatic metadata generation methods can be plugged into this framework. Applications, such as repositories that need to generate metadata, can access the framework for generating metadata. Dahl and Vossen propose a hybrid approach for metadata generation that is partially based on the Web 2.0 paradigm [DV07]. The creation of metadata is split between the author of a learning resource, the learner, and a platform. Author and learner manually create metadata; the platform applies automatic metadata creation methods. An overview over various human and automatic metadata generation methods is also given by Bergstraesser [Ber05].

## 6.2 A New Categorization Approach Based on Wikipedia as Substitute Corpus

As mentioned in the previous section, various machine learning methods exist. If it is intended to apply machine learning methods for metadata generation, several tasks have to be considered. Based on a particular scenario that is described below, a suitable method is proposed in this section. For that purpose, first a suitable training corpus has to be found. A measure for evaluation of the classification effectiveness is also required. Afterwards, a particular categorization method has to be chosen and tuned with suitable parameters. The stages of the whole process that leads to a classifier is depicted in Figure 6.2.

### 6.2.1 A Scenario for Domain-Independent Topical Metadata Generation

Learning object repositories (LOR) store learning resources for various purposes. With the large number of repositories that exist today, also the purposes of the repository, the contained learning resources, and the users differ. Some repositories are focussed on a narrow topic (e. g. GROW, the Geotechnical, Rock and Water Resources Library[25]), others contain a very broad range of subjects (e. g. MERLOT[26]).

In our scenario, we assume that a new repository for learning resources without domain restriction is established. That means, in the beginning there are no learning resources available in the repository. Because there are no domain restrictions, a predefined classification system for the expected learning resources does not exist. Two examples for such a kind of repository are a university-wide repository and an open marketplace for learning resources. In the first case it is assumed that a university wants

[25] http://www.grow.arizona.edu/
[26] http://www.merlot.org

to establish a new university-wide repository of academic learning resources. This repository shall be filled with learning resources by all faculties. It is not restricted to particular knowledge domains and there is no limitation of topics. In the beginning, the repository is almost empty; it grows over time. The range of contained topics might even change over time, when learning resources from new domains become available. That is, there is no a priori knowledge about the topics of learning resources that will be stored in the repository. The Content Sharing project[27] may serve as a second example. The Content Sharing marketplace provides a platform on which learning resources may be traded by producers and consumers. Here again, there are no learning resources available in the beginning; the marketplace repository is filled in the operating stage. Again, the topics of traded learning resources are not a priori known. Nonetheless, users expect to efficiently retrieve learning resources.

The repository in our scenario is supposed to support two retrieval tasks by suitable metadata: searching and browsing. Browsing over learning resources is different from browsing Web pages on the Internet. Web pages are hypertext documents, which contain lots of links to related Web pages, whereas learning resources should not contain references to other learning resources in order to remain self-contained. Hence, metadata has to be used for browsing. There are mainly two metadata fields in the LOM specification, which are predestinated for browsing: relations and classifications. Relations directly refer from the metadata record of one learning resource to another learning resource. However, relations are rarely specified by authors. An automated classification of learning resources could be accomplish, as machine learning methods can be applied (see Section 6.1). If learning resources are classified into hierarchical categories, browsing can be accomplished on two levels: Firstly, users may navigate from a learning resource to other learning resources within the same category. And secondly, the user may browse over the hierarchically organized category graph. With such a browsing approach, the user may start in the root category and descent into subcategories until he has found his area of interest. Or he begins at a known learning resource or category and explores the adjacent categories.

For this scenario, an automatic metadata generation method is helpful, which classifies learning resources into hierarchical categories. Furthermore, a hierarchical category system is needed, which covers a very broad range of topics. Such a category system can be used for both searching and browsing.

### 6.2.2 The Wikipedia-Based Categorization Approach

Corresponding to the scenario, a classifier is needed that works under the conditions of a large number of categories, a priori unknown topics, and an initially small amount of available learning resources. First, a training corpus has to be identified for training a machine learning classifier.

As the section on related work has shown, there are a lot of field-tested methods for categorization of documents in general. Learning resources have likewise been categorized successfully in the past, for instance by Saini et al. [SRS06]. However, these methods are applied only to a small number of categories – 41 in the case of Saini. A major challenge arises from the fact that all known machine learning algorithms require several positive and optionally also negative sample documents per category for learning a classifier. Consequently, a large number of categories demands for an even larger number

[27] http://www.contentsharing.com

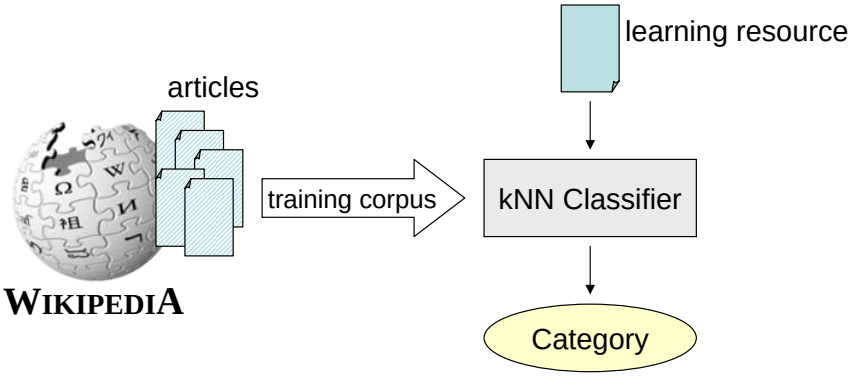


Figure 6.3: Using Wikipedia articles as substitute corpus.

of already labeled learning resources as training corpus. In our scenario, this training corpus does not exist in the repository, because only few learning resources are present in the beginning.

To solve this challenge, we propose to replace the conventional training corpus of a machine learning method by a substitute corpus. As a substitute corpus, a set of document has to be found, which fulfills two criteria:

1. The documents have to be similar to the learning resources that are to be classified later. Similarity in this regard means that a machine learning method can infer from substitute documents to real learning resources[28] afterwards.

2. The sample documents have to be already labeled by concepts from a large-scale thematic taxonomy. It is assumed that it is not feasibly to manually label a large number of sample documents because of economic reasons.

A corpus that might fulfill these two criteria is the free encyclopedia Wikipedia[29]. Wikipedia contains a huge number of encyclopedic articles about various topics. If there is a topic that is important for a particular community, it is as well likely that a Wikipedia article about that topic exists. Furthermore, several instances of Wikipedia for different languages exist, for instance the German Wikipedia[30]. As of June 2008, the English Wikipedia contains about 2,400,000 articles; the German Wikipedia accounts for 762,000 articles at that time. Wikipedia articles are organized in thematic categories. There are, for instance, more than 50,000 categories in the German Wikipedia. A category may be part of a more generic category. All categories form a directed acyclic graph[31]; they have a common root category. Ponzetto and Strube have compared the Wikipedia category systems to two other taxonomies:

> "We compared the created taxonomy with ResearchCyc and via semantic similarity measures with WordNet. Our Wikipedia-based taxonomy proved to be competitive with the two arguably largest and best developed existing ontologies." [PS07]

[28] The term *real learning resource* is used to identify those learning resources, which are stored in the repository from our scenario.
[29] http://en.wikipedia.org
[30] http://de.wikipedia.org
[31] Actually, sometimes cycles have been detected and removed by Wikipedia administrators in the past. We assume that the category system now is a directed acyclic graph.

That analysis supports the usage of Wikipedia categories as taxonomy. There are some domain-specific taxonomies, which possibly describe a particular domain in more detail. For instance, the Medical Subject Headings (MeSH)[32] thesaurus exists for medical resources. But these taxonomies are applicable only for that limited domain. The Wikipedia category system seems to be well suited for the given domain-independent scenario.

The proposed approach for a training corpus is to use Wikipedia articles as substitute documents. The categories to which an article is assigned serve as labels. A machine learning method is used to learn classifiers for all Wikipedia categories. These classifiers should afterwards be able to correctly classify real learning resources (figure 6.3). This approach is experimentally evaluated in the next two sections. Existing approaches do not use Wikipedia articles as training documents for machine learning methods. Thus, the Wikipedia-based classifier is a novel approach.

### 6.2.3 Related Work Using Wikipedia as Knowledge Source

Many researchers have identified Wikipedia as a rich source of knowledge for automatic processing before[33]. The Wikipedia corpus is used for different purposes, for instance for ontology extraction, for extracting linguistic knowledge, or for developing or improving classifiers. There are also works that research the development and growth of the Wikipedia [Vos05].

Voss explains the category system of Wikipedia and compares it to other indexing types. He points out that the Wikipedia category system is designed as a thesaurus [Vos06]. Voss also describes how categories are technically stored and linked to each other. Ponzetto and Strube extract the category system of Wikpedia as a large scale taxonomy [PS07]. They identify the advantages of the Wikipedia category system as domain independent, up-to-date, and multilingual. In order to evaluate the quality of this taxonomy, the coverage and quality of semantic relations are compared to other taxonomies. YAGO is another approach for extracting ontologies out of Wikipedia [SKW07]. YAGO extracts and combines knowledge from Wikipedia and WordNet [Mil95] to build a large, light-weight ontology.

Besides the extraction of ontologies, the Wikipedia corpus serves also as a source of linguistic knowledge. This kind of knowledge is frequently used as input to further improve other text processing, machine learning and information retrieval methods. For instance, Strube and Ponzetto extract information about the pairwise relatedness of words from Wikipedia [SP06]. Wikipedia proves to be a better information source for this task than a baseline that uses the Google search engine. The community-generated Wikipedia is even comparable to hand-crafted taxonomies, such as WordNet. In [LGFM08] we use Wikipedia pages for the disambiguation of terms.

There are some approaches that retrieve other general purpose category systems than Wikipedia as initial source for categorization tasks. Chekuri et al. retrieve categorized Web documents from the Yahoo Web catalog. These documents are used to learn a classifier for categorizing new documents into these categories [CGRU97]. Not the whole Yahoo taxonomy is used, but only 20 high-level categories and 2000 training documents. LiveClassifier is a system for automatically collecting corpora from the Internet

[32] http://www.nlm.nih.gov/mesh/

[33] An incomplete list of academic studies that use Wikipedia as knowledge source exists as a Wikipedia project page: http://en.wikipedia.org/wiki/Wikipedia:Wikipedia_in_academic_studies.

[HCC04]. Requiring only a user-defined topic hierarchy, the system retrieves a corpus from a Web search engine and trains a classifier with this corpus.

The approach of Schönhofen is similar to our method presented in this section: he uses Wikipedia articles and categories to determine the topics of documents [Sch06]. The two approaches differ insofar that Schönhofen uses only article and category titles to find category candidates for a document, but not the text of Wikipedia articles. First, he analyzed how good the categories for articles taken from Wikipedia are predicted. Afterwards, he uses predicted categories as features to improve the categorization of newsgroups postings. The work of Syed et al. employs Wikipedia article names and categories to describe the common concepts of a set of documents; they compare documents to the full text of articles using the cosine similarity [SFJ07]. The most similar articles for each document of a given set are used to calculate the common concepts of that document set.

Gabrilovich et al. [GM07a] use the Wikipedia corpus for feature generation in order to improve machine learning methods. Text documents are transformed into concept vectors. Each dimension of a concept vector is represented by a Wikipedia article; the corresponding values express how related a given document is to a particular article. This approach is called Explicit Semantic Analysis (ESA). Concept vectors are used instead of the original word vectors as input for text categorization systems. Yet another kind of information is used by Potthast et al. [PSA08]. Wikipedia articles may contain links to an equivalent article in another language instance of Wikipedia. Potthast et al. make use of these links for cross-language retrieval.

### 6.2.4 Choosing Effectiveness Measures for Hierarchical Categorization

Precision and recall are based on binary decisions; that means, a determined category is either correct or false. However, this binary decision is not adequate if hierarchical categories are used [MPL06]. As an example, consider that an object, which belongs to the category *Bacteria* is spuriously assigned to *Bacterial diseases*. This classification is wrong, but on the other hand not as bad as if the object had been classified as *Cars*.

Instead of using binary decisions, several evaluation methods use scalar values to rate the misclassification. All these methods have in common that they calculate a value for the pair of correct and determined category, which can be referred to as similarity, distance or misclassification costs. Resnik calculates a so called information content for concept nodes of the WordNet taxonomy [Res95]. The semantic similarity of two concepts is then computed as the maximum information content value of all concepts that subsume the two concepts. Resnik's evaluation shows that using information content outperforms measures that count the path length between two concepts. Sun and Lim calculate the contribution of a document to a particular category based on the cosine of feature vectors [SL01]. Extended precision and recall measures for a category are determined afterwards by taking all contributions to this category into account. It has been shown that these extended performance measure can improve hierarchical classification [SLN03].

Resnik's notion of information content requires a corpus, in which the frequency of concepts can be counted. Seco et al. propose a new method for determining the information content of concepts, which analyzes only the structure of the taxonomy [SVH04]. Strube and Ponzetto found out that for

the categories of Wikipedia this method correlates better with human judgement than Resnik's method [SP06].

As Strube and Ponzetto have successfully tested their measure on the Wikipedia corpus, this measure is adopted for this thesis. It counts the hyponyms of a category and compares it to the overall number of categories in the taxonomy. Given that $hyponyms(c)$ is the number of subcategories subsumed by a category $c$ and $|C|$ is the number of all categories, the information content of a category $c$ is then calculated by equation 6.1.

$$ic(c) = 1 - \frac{log(hyponyms(c)+1)}{log(|C|)} \tag{6.1}$$

To calculate the similarity of two categories $c_1$ and $c_2$, Resnik has defined a similarity function $sim_{res}$ as follows [Res95]:

$$sim_{res}(c_1, c_2) = \max_{c \in S(c_1, c_2)} ic(c) \tag{6.2}$$

$S(c_1, c_2)$ is the set of all categories that subsume $c_1$ and $c_2$.

Lin has analyzed different similarity functions and defined a theoretical foundation [Lin98]. Seco et al. have applied one of Lin's formulas to information content. This equation can also be transformed into a distance function, which is equivalent to the definition of Jiang and Conrath [JC97] (cp. equation 6.3). Note that the equation has been normalized in order to obtain values between 0 and 1.

$$dist_{jcn}(c_1, c_2) = \frac{(ic(c_1) + ic(c_2)) - 2 * sim_{res}(c_1, c_2)}{2} \tag{6.3}$$

Knowing the distance between two categories, we can now calculate performance measures, such as precision, recall and accuracy. In our scenario the overall performance and not the performance per category is of interest; therefore the overall accuracy is determined. The accuracy calculation is based on the cost-based accuracy, as defined by Li and Bontcheva [LB07]; the distance from equation 6.3 is used as cost function. According to [LB07] the hierarchical accuracy $acc_{hier}$ is given in equation 6.4, where $c_i \in C_M$ is the determined category of learning resource $i$ using method $M$ and $l_i \in L$ is the corresponding reference label, which is considered as ground truth.

$$acc_{hier}(C_M, L) = \frac{\sum_{i=1}^{n}(1 - dist_{jcn}(c_i, l_i))}{n} \tag{6.4}$$

For measuring the performance of different categorizations in the following sections, two performance functions are used. One is the flat accuracy $acc_{flat}$, which counts how many learning resources have been

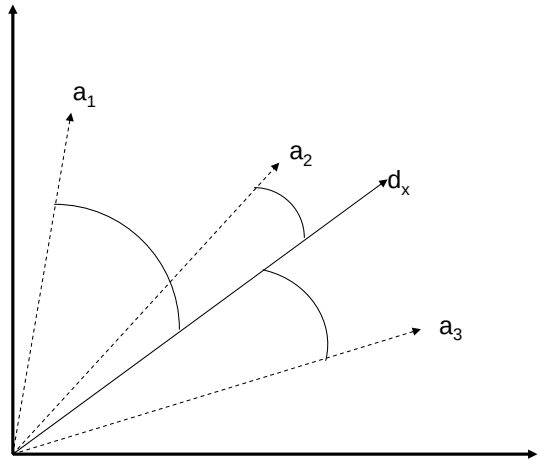


Figure 6.4: Comparison of documents in a document vector space.

categorized into the correct category. The hierarchical accuracy $acc_{hier}$ is used as second measure to also take different degrees of misclassification into account.

$$acc_{flat}(C_M, L) = \frac{|\{i|c_i = l_i, 1 \leq i \leq n\}|}{n} \tag{6.5}$$

### 6.2.5 Implementation of the New Classifier

In order to evaluate whether the proposed approach is feasible, the categorization method has been implemented. This subsection describes which machine learning method has been selected, how the actual classifier has been implemented, how the Wikipedia corpus has been prepared for the method, and which test data that has been used for evaluation.

#### A Wikipedia-based Classifier Using the k-Nearest-Neighbor Method

Before a classifier can be implemented a decision for one categorization method has to be made. There are several methods known from machine learning, which could be applicable, for instance Bayes classifiers, decision trees, support vector machines, k-nearest-neighbors (kNN), and many more. All of these methods basically perform the same task, but with a different effectiveness and efficiency [Yan99].

The given scenario requires learning to classify a large number of categories: for the German Wikipedia instance more than 40,000 categories, using several hundreds of thousands of articles as training documents. At this order of magnitude, the efficiency of a categorization method (time and memory resources required for training and classification) has to be considered. The kNN method has been chosen for three reasons: Firstly, it is known to work well with large amounts of classes [Yan99]. Secondly, the kNN method does not require an off-line training phase; this probably makes the comparison of different parameter settings faster. And finally, the kNN method can be extended by additional information, such as links between articles or categories. Other categorization methods, such as support vector machines

or Bayes classifiers would require a separate classifier for each category and furthermore a long training phase; for the large number of 40,000 categories the resources of a typical workstation are supposed to be insufficient.

Single-label categorization has been chosen for the evaluation. This means that each learning resource is assigned to exactly one category. A learning resource often contains aspects that could belong to different categories; we assign though a learning resource only to the one category, which subsumes the contents best. A multi-label approach could also be applied and would probably be better suited for real-world applications. However, a multi-label approach would bring up two challenges: Firstly, the manual labeling of the sample learning resources becomes more difficult, because it is unclear how much contents of a learning resource have to be related to a category for making the learning resource a member of that category. For instance, does a learning resource about diesel fuel besides the category *fuels* also belong to the categories *diesel engines*, *German inventors* or *petroleum products*? Furthermore, the categorization method would involve additional parameters that have to be tuned in order to achieve an acceptable effectiveness.

The k-nearest-neighbors method requires that documents are transformed into feature vectors. For a given query document $d_x$ the document's feature vector $\vec{d_x}$ is compared to the feature vectors of the training documents. The comparison may use for example the Euclidean distance or the cosine measure, which is based on the angle between two vectors (see Figure 6.4). Let $A$ be the set of all articles of Wikipedia in a chosen language and $T$ the set of all terms (words) that occur in the Wikipedia corpus. The terms in $T$, or a subset $T^*$ thereof, are used as features. Thus, the elements of $T^*$ span the document vector space in which documents are compared. The document similarity $\sigma$ between the document $d_x$ and a Wikipedia article $a_i \in A$ is, for instance, defined as the cosine similarity.

$$\sigma(d_x, a_i) := \sigma_{cos}(d_x, a_i) = \frac{\vec{d_x} * \vec{a_i}}{|\vec{d_x}| * |\vec{a_i}|} \tag{6.6}$$

Categorization of a document with the kNN method is based on the similarity of documents. For the document $d_x$ the $k$ most similar articles (neighbors, $n(d_x)$) are determined; the number $k$ is specified as parameter. For each article we know to which categories it belongs. The occurrences of categories of articles in $n(d_x)$ are counted. The document $d_x$ is assigned to the category $c_y \in C$ to which most of the $k$ articles belong.

$$n(d_x) = \{a_i \in A, \quad |\{a_j \in A | \sigma(d_x, a_j) > \sigma(d_x, a_i)\}| < k\} \tag{6.7}$$

The influence of an article on the category to determine can be either real valued, depending on the distance between $d_x$ and the article, or discrete [Mit97]. The discrete-valued function has been chosen for the present implementation, because it is easier to calculate and the impact of distance weights is

assumed to be marginal. The rank $r$ of a category $c$ regarding the document $d_x$ is specified as the number of neighbor articles that belong to $c$.

$$r_{d_x}(c_j) = |\{a_i \in n(d_x) \mid a_i \in c_j\}| \tag{6.8}$$

Finally, the document $d_x$ is assigned to the category $c_y$ that has the highest rank among the neighbors.

$$cat_{kNN}(d_x) := c_y \in C \quad with \quad r_{d_x}(c_y) = max\{r_{d_x}(c_j) \mid c_j \in C\} \tag{6.9}$$

**Algorithm 6.2.1:** WIKIPEDIABASEDCATEGORIZATION($LR$)

```
transform learning resource LR into a vector representation
for each article A[i] ∈ WIKIPEDIA
  do calculate similarity σ(LR,A[i])
sort articles by decreasing σ
S ← top K articles with highest σ
CAT ← category that is most frequent in S
return (CAT)
```

The overall classification algorithm is also given as pseudocode in Algorithm 6.2.1. As mentioned above there are some parameters that influence the selection of neighbors and thus also the categorization result. For instance, the value $k$ has a high impact. Suitable parameters have to be determined empirically. This parameter calibration is done in Section 6.3.

### Parameters of the Wikipedia-Based Classifier

The effectiveness of a k-nearest-neighbors method depends strongly on the choice of certain parameters. Lim shows that "it is very worthy of tuning parameters of kNN method to increase performance rather than having hard time in developing a new learning method" [Lim04]. Following this finding, the method proposed in this chapter was calibrated by optimizing a number of parameters (see Figure 6.5). Different parameters and variations of the kNN method have been tested. They can be divided into two groups. The first group are parameters that influence the calculation of similarity between two documents. The second group subsumes parameters and method variants that determine how from document similarities the category membership of a learning resource is inferred. Group one comprises dimensionality reduction and the selection of a similarity measure for feature vectors. Group two consists of the value $k$ and of various approaches to replace the basic kNN method by algorithms that exploit knowledge about the

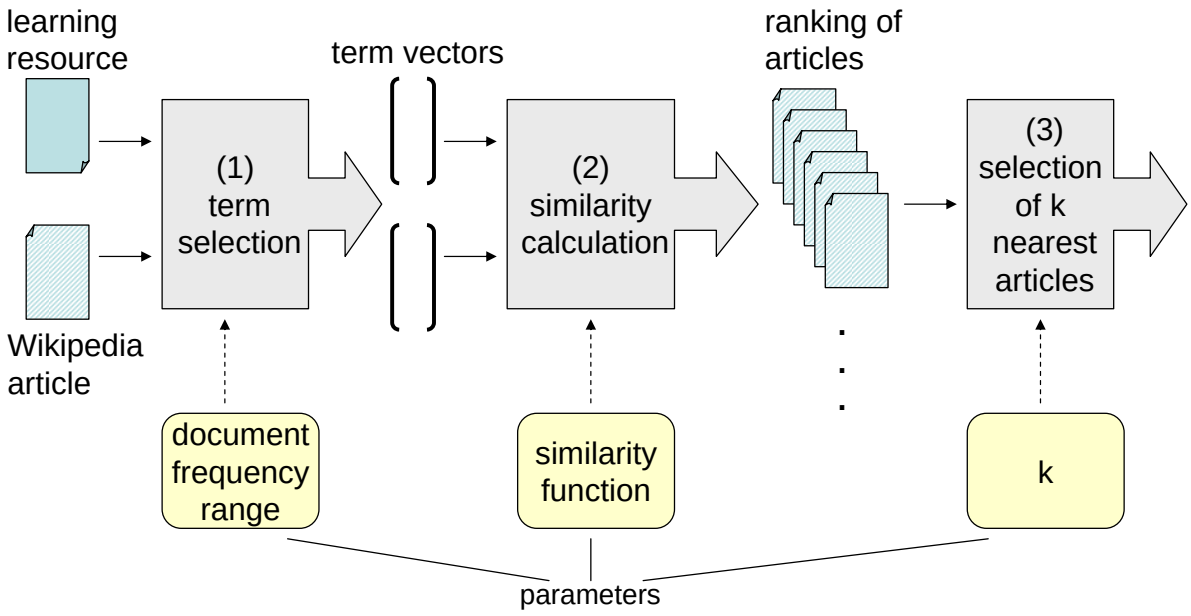


Figure 6.5: Parameters of the Wikipedia-based kNN classifier.

internal hierarchy of a learning resource, the category system structure, or the page link structure of the Wikipedia corpus.

As measures for computing the similarity of two vectors in the word vector space, the cosine measure, the Jaccard coefficient and an overlap coefficient were evaluated[34]. These measures will be formally specified in the next section.

Sebastiani considers dimensionality reduction as a means to improve the effectiveness and efficiency of machine learning methods [Seb02]. Methods for dimensionality reduction are distinguished into term selection methods and term extraction methods. Term selection means that a subset of the original set of terms is used for classification. Term extraction methods generate new terms that are not elements of the original set of terms; exemplary methods for term extraction are latent semantic indexing and random indexing [Sah05]. Term selection methods employ different metrics for determining which terms are most relevant for the categorization quality. One metric is the document frequency of terms that measures in how many different documents of a given corpus a particular term occurs. Yang and Pedersen point out that the document frequency of terms is comparable to other, more complex measures [YP97]. Thus, term selection is based on document frequency as measure in this thesis.

The basic implementation of the categorization method used a traditional discrete-valued method. A set of $k$ most similar articles for a given query document $d_x$ was determined. The category to which most of these neighbors belong was chosen as category of $d_x$. In addition, two variants of the basic methods were developed and tested that tried to improve the Wikipedia-based classifier in a late phase; the prediction function of a category from a set of determined similar articles was modified. The first variant used the hierarchical structure of the category system for rating also more generic categories of an article. The second variant decomposed a learning resource into several fragments and determines

[34] In the first experiments the Dice coefficient was implemented as a fourth similarity measure [MRS07]. However, as the results of the Dice coefficient were equal to those of the Jaccard coefficient the Dice coefficient has been omitted in the diagrams and tables in this chapter for reasons of clarity.

neighbors for each fragment. The two variants and their impact on the categorization performance are described in the next section in more detail.

### Preparation of the Wikipedia Corpus

Using the kNN method for categorization requires to compare documents in a word vector space. If all words that occur in the Wikipedia corpus are used as base vectors the dimensionality of the vector space may reach about three million. Hence, each document would be represented by a vector with three million elements. Considering also that the Wikipedia corpus contains several hundred thousand articles, the memory requirements and the computing time for all similarities may be immense [MRS08]. Efficient storage forms and algorithms have to be applied.

The uncompressed database dumps of Wikipedia instances have a size between 2 GB (German Wikipedia) and 5 GB (English Wikipedia). If the classifier should fit completely into main memory (reading data from hard disk is an expensive operation) the reduction of memory consumption has a high priority. First of all, a sparse vector representation is used, which means that only non-zero elements of vectors are stored. Considering, that most articles contain only some hundred different words out of the vocabulary of 3 million words, the effect is significant. Additionally, the dimensionality of the word vector space was reduced (see below); however, sparse vectors are still necessary.

The Wikipedia corpus was preprocessed in order to enable efficient calculation of similarity values when a document is to be classified. A database dump of the German Wikipedia was downloaded from the Internet in February 2007 as XML file [35]. First, the total set of Wikipedia pages has been filtered for regular articles. Wikipedia contains different types of pages, such as regular articles, category pages, Wikipedia project pages, image pages, and so on. Only regular articles were used as reference documents for the categorization method. From these regular articles, the pure texts were extracted; markup was discarded.

The run-time representation of article vectors was realized using the compressed sparse row format, which contains only none-zero values plus two index vectors: a column index, which determines the position within a vector, and a row index which indicates where each vector starts [GJS03]. An additional hash table for fast random access to non-zero values was introduced as extension to the compressed sparse row. Based on the compressed sparse row representation for article vectors and a hash table representation of learning resource vectors, an optimized algorithm for calculating the similarity was implemented.

For a hierarchical evaluation as specified above the structure of the category graph and the category membership of articles have to be known. This information was also extracted from the Wikipedia corpus. Two graphs were extracted: a category graph that contains all links from any page to a category; and a page link graph that contains all links between any two Wikipedia pages. Both graphs are stored as a set of identifier pairs, where each identifier pair consists of the source and the destination of a link.

[35] Database dumps are available as XML and SQL dumps for different languages at http://download.wikimedia.org/.

### Sample Learning Resources

Obtaining useful sample data is crucial for the evaluation of a categorization method. In the general area of information retrieval, researchers use common corpora (e. g. the TREC collections[36] or Reuters news corpora [LYRL04]) in order to ensure that results are comparable to other approaches. A suitable sample corpus of learning resources is to the best of our knowledge not existing at the moment. A procedure to obtain sample learning resources could be to gather them manually by searching on the Internet [SRS06] or to automatically retrieve Web resources based on specified category labels [HCC04]. However, we believe that an evaluation is most valuable if authentic learning resources are used, which are applied for teaching in practice.

*Knowledge in Medical Education (k-MED)* is an interdisciplinary project that supports e-learning in medical education. WBT learning resources about various medical topics are created, used and shared by teachers at several German universities. The learning resources are created in the ResourceCenter authoring environment [HHRS05] and transformed into SCORM packages for learning. There are currently more than 200 learning resources available at package level granularity. While this number of learning resources is insufficient for training a classifier, it is large enough for calibration and an overall evaluation. 150 packages have been downloaded from the k-MED repository as samples for the evaluation of the Wikipedia-based classifier; all of them are written in German language. The 150 learning resources have been split into two sets: 100 learning resources form a calibration set for determining suitable parameters for the method; the remaining 50 learning resources are used as a test set. In the terminology of machine learning, a *training set* is required to build a classifier; in the case of the proposed method, the Wikipedia corpus serves as a training set. Each of the k-MED learning resources is manually assigned to the Wikipedia category where it is supposed to best fit in; these manual labels are regarded as ground truth. As a preparation for processing, the pure texts have been extracted from each SCORM package. Additionally, the internal hierarchical structure of these texts has been extracted for certain extensions of the categorization method.

## 6.3 Experimental Calibration of the Wikipedia-Based Classifier

Section 6.2 has presented a new approach for categorizing learning resources. This approach replaces the typical training corpus by Wikipedia articles as substitutes. The method was implemented and applied to a set of real learning resources. Different values for a number of parameters were varied in order to calibrate the categorization method. The results of the calibration are presented in this section.

### 6.3.1 Experiment 1 – The Basic Wikipedia-Based Classifier

In a first experiment a discrete-valued k-nearest-neighbors classifier is implemented. The classification task is to assign a given learning resource $d_x$ to exactly one category. The learning resource $d_x$ and the articles of the Wikipedia are transformed into term vectors. All words that occur in the Wikipedia corpus are initially considered as valid terms. A stemming algorithm is applied to merge different forms of the

[36] http://trec.nist.gov/data.html

same word stem[37]. The occurrence of words in $d_x$ are counted and weighted by the TF-IDF weighting scheme [SWY75]. TF-IDF means that the term frequency (how often a term occurs in the current document) is divided by the logarithm of the document frequency (in how many documents of the corpus the term occurs). Thus, terms that appear in only few documents are weighted higher than terms that are very common. The vectors are normalized to a length of 1.0 for facilitating further calculations. For computing the similarity between a query document vector $\vec{d_x}$ and an article vector $\vec{a_i}$ three different similarity measures were applied. The cosine similarity is defined as inner product of two vectors.

$$ds_{cos}(d_x, a_i) = \frac{\vec{d_x} * \vec{a_i}}{|\vec{d_x}| * |\vec{a_i}|} = \vec{d_x} * \vec{a_i} \tag{6.10}$$

The remaining two similarity functions operate on term sets instead of vectors. Let $X$ be the set of terms occurring in $d_x$, and $Y_i$ the set of terms in $a_i$. Then, the Jaccard similarity and the overlap coefficient are calculated as set operations:

$$ds_{Jaccard}(d_x, a_i) = \frac{|X \cap Y_i|}{|X| + |Y| - |X \cap Y_i|} \tag{6.11}$$

$$ds_{Overlap}(d_x, a_i) = \frac{|X \cap Y_i|}{min(|X|, |Y_i|)} \tag{6.12}$$

The second parameter that has been varied is term selection. Lim shows that the kNN method may achieve good result even if the number of terms is reduced by 90% if the right terms are selected [Lim04]. Yang and Pedersen prove that the document frequency is a suitable criterion for term selection [YP97]. Therefore, document frequency (DF) was chosen as selection criterion. Terms with high and low frequencies were removed. At the low DF end, terms were removed that occurred in only one or two documents; but also a measurement including these terms was performed. At the other end, document frequencies of 50,000, 100,000 and 200,000 were used as upper limit; again, having no upper limit was an option. In total, 7 combinations of low and high DF limits were compared.

The articles were ranked by their similarity with $d_x$. The $k$ most similar articles were used for determining the category membership of $d_x$. Different values for $k$ (1, 3, 5, 10, 20, 30) were compared for calibration.

The effectiveness of the Wikipedia-based classifier for combinations of the three parameters (similarity measure, term selection, $k$) was measured by applying the classifier to a calibration set of 100 learning resources. Two different measures were used. First, the flat accuracy $acc_{flat}$ has been calculated by analyzing how many learning resources have been classified into the exactly correct category out of about 40,000. As a second measure, the hierarchical accuracy $acc_{hier}$ also regards how far the distance between a mismatching category and the correct one is (cp. Section 6.2.4).

[37] The German stemming algorithm form the Snowball project (http://snowball.tartarus.org/) has been used.

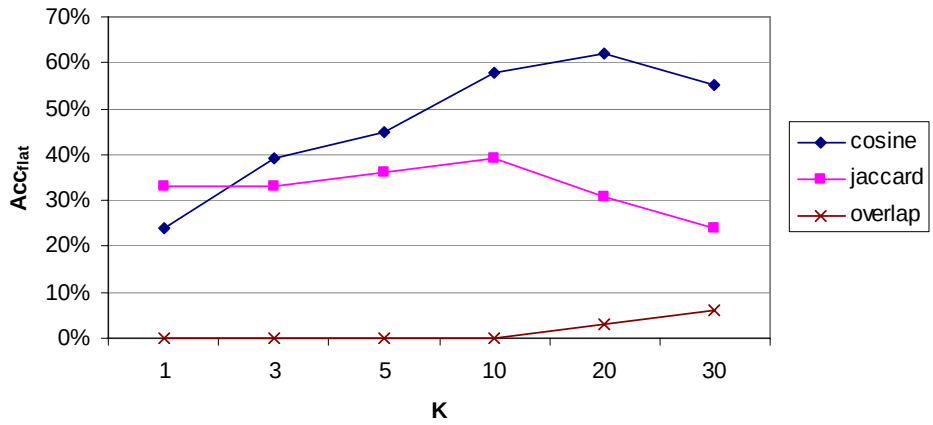


Figure 6.6: Comparison of flat accuracy of the three similarity measures for a fixed term selection ($3 \leq DF \leq 100,000$).

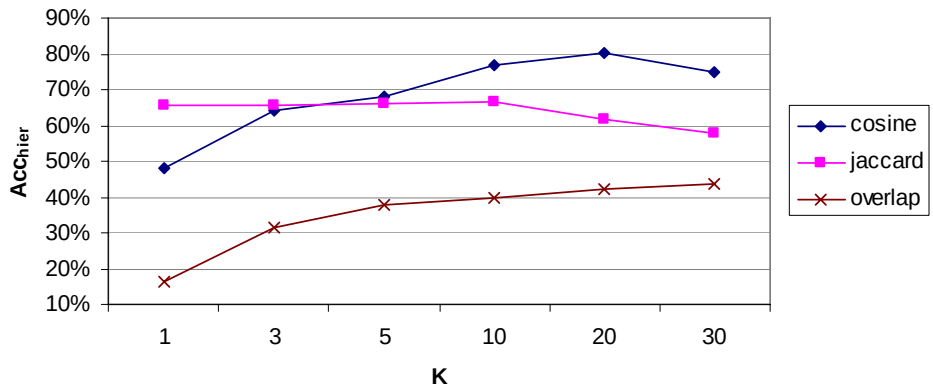


Figure 6.7: Comparison of hierarchical accuracy of the three similarity measures for a fixed term selection ($3 \leq DF \leq 100,000$).

Table 6.1: Accuracy measurements ($acc_{flat}/acc_{hier}$) for a fixed term selection ($3 \leq DF \leq 100,000$).

| $k$ | **cosine** | **jaccard** | **overlap** |
|---|---|---|---|
| 1 | 24% / 48,20% | 33% / 65,51% | 0% / 16,28% |
| 3 | 39% / 64,26% | 33% / 65,66% | 0% / 31,70% |
| 5 | 45% / 68,03% | 36% / 66,11% | 0% / 37,88% |
| 10 | 58% / 76,81% | 39% / 66,40% | 0% / 39,86% |
| 20 | 62% / 80,34% | 31% / 61,71% | 3% / 42,13% |
| 30 | 55% / 74,96% | 24% / 57,97% | 6% / 43,47% |

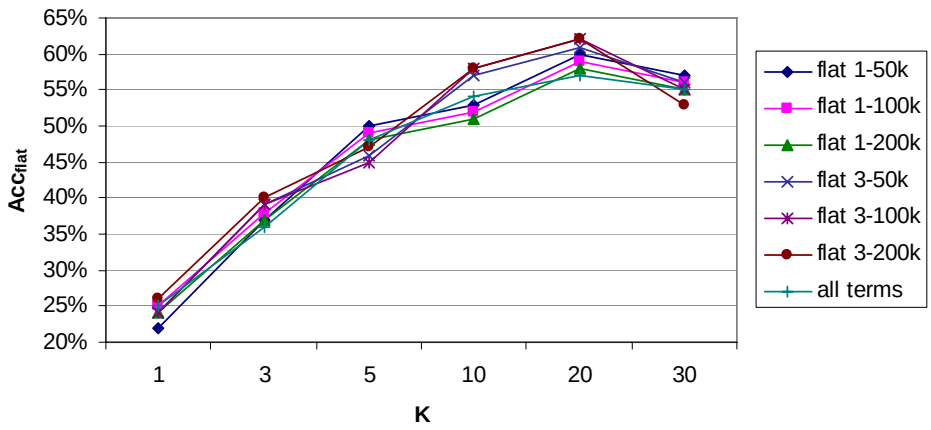


Figure 6.8: Comparison of different term selections with the cosine measure (flat accuracy).

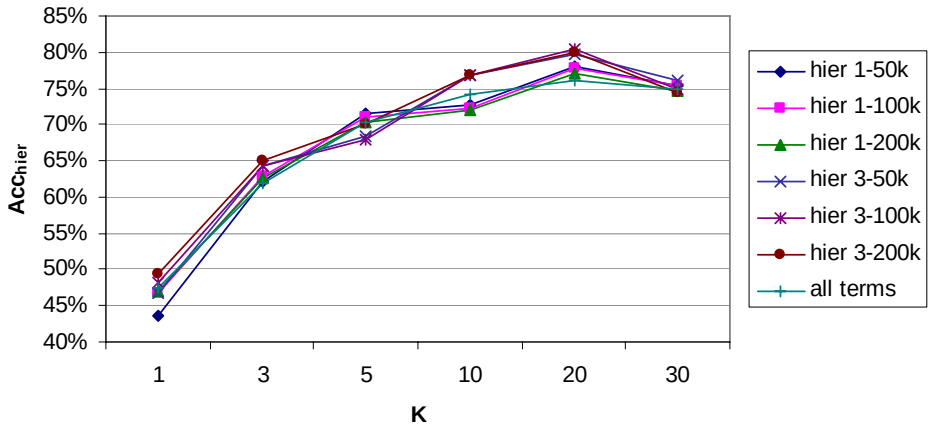


Figure 6.9: Comparison of different term selections with the cosine measure (hierarchical accuracy).

As the categorization has been performed with three variable parameters, the results were analyzed individually for each of the parameters. First, the three implemented similarity measures (cosine, Jaccard and overlap) are compared. For visualizing the performance of these measures, a term selection of document frequencies between 3 and 100,000 is fixed. The $k$ value remains as free parameter. The achieved flat and hierarchical accuracy for each of the three similarity measures for different values of $k$ are illustrated in Figures 6.6 and 6.7. The measurements for these diagrams are also listed in Table 6.1. What can be seen in the two diagrams is that the cosine measure significantly outperforms the other two similarity measures if a good value for $k$ is chosen. The overlap measure is by far the worst measure. The performance of the Jaccard measure is less strongly affected by the choice of $k$ than the cosine similarity. This observation is confirmed by the diagrams for other term selections (see Appendix B).

Next, the impact of term selection is regarded. For this purpose, only the cosine similarity is used as similarity measure. The accuracy curves for different term selections over varying values of $k$ are compared in the Figures 6.8 and 6.9. All seven curves are very close to each other. This means that the choice of $k$ has a much higher impact than the choice of a term selection. A $k$ of 20 generates the best accuracy values for both flat and hierarchical accuracy. Nonetheless, there are performance variations for different term selections. Two characteristics can be observed. First, removing terms with a low document frequency (less than three documents) always improved the accuracy in the experiment. Reducing high

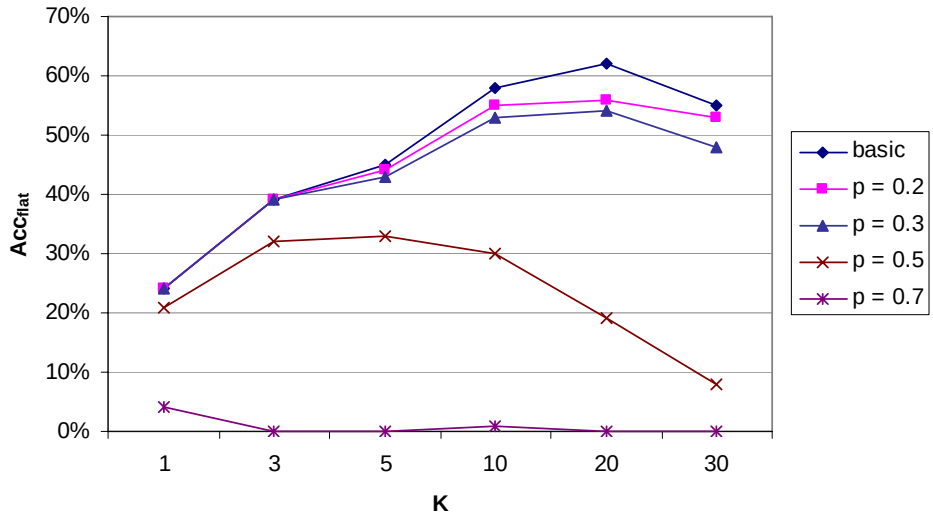


Figure 6.10: Flat accuracy measurement for hierarchical category propagation ($3 \leq DF \leq 100,000$).

document frequencies down to 100,000 increases the effectiveness; a further reduction to 50,000 leads to suboptimal results. Thus, a term selection of document frequencies between 3 and 100,000 can be interpreted as optimal with respect to the given set of documents.

Overall, a set of optimal parameter settings has been determined. These settings are:

- cosine measure as similarity measure
- term selection: $3 \leq DF \leq 100,000$
- $k = 20$

The implemented Wikipedia-based classifier in combination with the determined parameter values are further referred to as *basic method*. Following subsections will describe variations of the method and compare them to the basic method. For now, the determined parameters are only valid for the used training set of learning resources. In order to show that the parameters are also valid for the general case, they have to be confirmed by applying the classifier to the test set.

### 6.3.2 Experiment 2 – Generalization by Hierarchical Propagation

The basic method considers only the immediate categories of related articles as candidates. Measured hierarchical accuracy values of 80% indicate that falsely assigned categories are not completely wrong. The assigned category is often at least similar to the ground truth category. A modified version of the basic method tries to leverage the potential of these nearby misclassifications. The idea is to introduce a mechanism of generalization. Generalization is realized by propagation of category relevance to more general categories of an article. As an example, we assume that the set of most similar articles for a learning resource consists of 10 articles from the category *Viral diseases* and 10 articles from the category *Bacterial diseases*; in this case, a decision for one of the two categories is likely to be wrong. Instead of considering only these two categories, the category *Infectious diseases* (which subsumes *Viral diseases* and

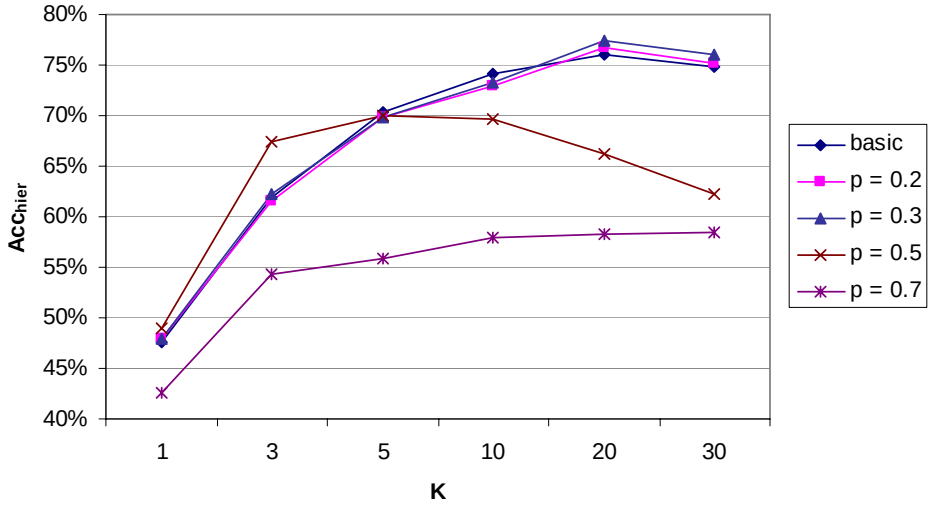


Figure 6.11: Hierarchical accuracy measurement for hierarchical category propagation (no term selection).

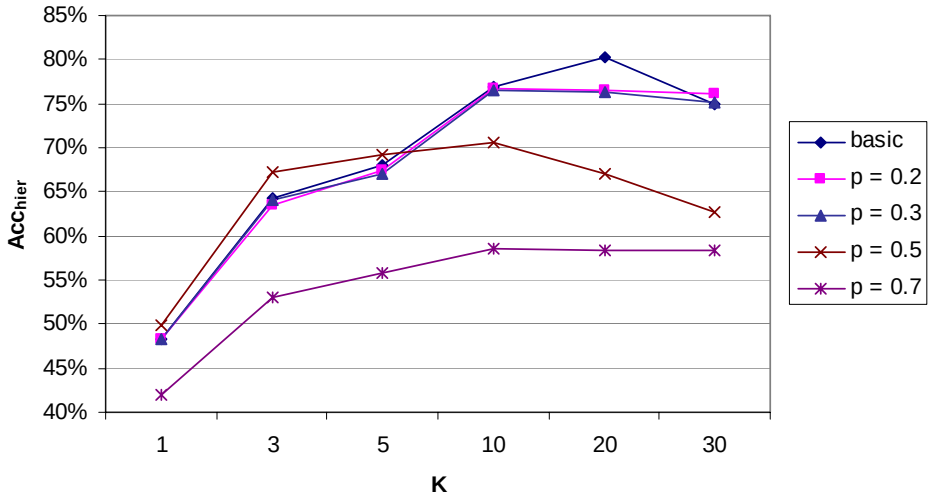


Figure 6.12: Hierarchical accuracy measurement for hierarchical category propagation ($3 \leq DF \leq 100,000$).

*Bacterial diseases*) could also be considered as a candidate. This propagation can be recursively repeated by also generalizing from *Infectious diseases* to *Diseases*, and so on.

However, this approach bears the risk of over-generalization: if predictions are more general than necessary, the information content of these predictions decreases. The experiment in this subsection should identify whether the utilization of hierarchical information results in a positive or negative effect on the classification effectiveness.

**Algorithm 6.3.1:** CATEGORIZATIONWITHHIERARCHICALPROPAGATION($LR$)

```
procedure VOTE(Category, Weight)
  RankingIndicator[Category] ← RankingIndicator[Category] + Weight
  SC ← all subcategories of Category
  for each C ∈ SC
    do VOTE(C, P * Weight)
  exit

main
  transform learning resource LR into a vector representation
  for each article A[i] ∈ WIKIPEDIA
    do calculate similarity σ(LR, A[i])
  sort articles by decreasing σ
  S ← top K articles with highest σ
  for each A ∈ S
    do for each category label C of A
    do VOTE(C, 1.0)
  CAT ← category with highest RankingIndicator
  return (CAT)
```

In order to utilize hierarchical information, also super categories of determined articles are taken into consideration. The strength of a category depends on the degree of generalization: the more general a category is in relation to a determined article, the less likely the category becomes a candidate[38]. A propagation factor $p$ specifies how strong the effects of more general categories are. Let $g$ be a super category of a category $c_i$. Then the influence $w(g)$ of $g$ on the classification is $w(g) = p * w(c_i)$. The influence of a category $c_a$, to which an article $a$ directly belongs is defined as $w(c_a) = 1$. This modified method is described as pseudocode on Algorithm 6.3.1.

This modified classifier has also been implemented and evaluated. Four propagation rates (0.2, 0.3, 0.5 and 0.7) have been compared to the basic method. First of all, the cosine measure again was superior to

[38] It is obvious that otherwise the root category would always be selected, because all articles recursively belong to that category.

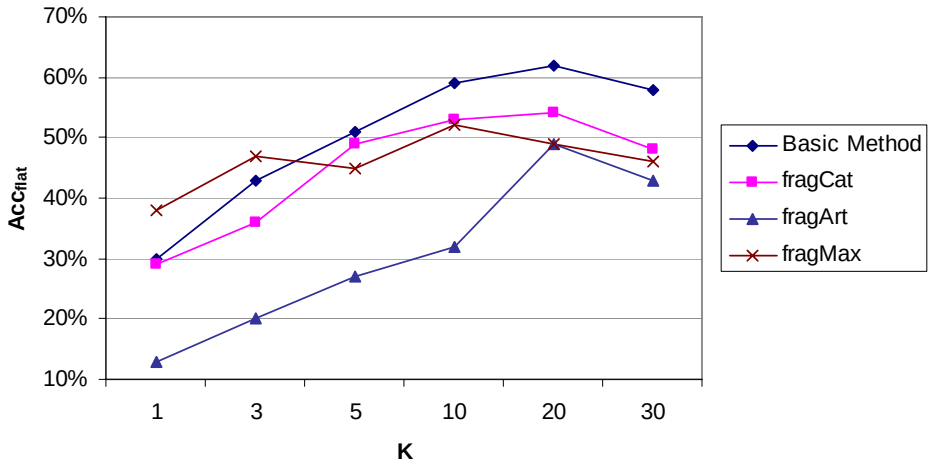


Figure 6.13: Comparison of fragment-based categorization (flat accuracy).

the other two similarity measures; hence, the discussion of results is focussed on outcomes of the cosine similarity function. Figure 6.10 illustrates how the hierarchical propagation method performs against the basic method with regard to the flat accuracy measure and given a term selection of $3 \leq DF \leq 100,000$[39]. The resulting curves show that hierarchical propagation cannot compete with the basic method when flat accuracy is concerned. Propagation rates of 0.2 and 0.3 create similar curves as the basic method (but still suboptimal). Propagation rates of 0.5 and 0.7 appear to be by far unsuitable.

In the case of hierarchical accuracy, the hierarchical propagation can keep up with the basic method (for propagation rates of 0.2 and 0.3). In a setting without term selection, the hierarchical propagation does even slightly better than the basic method (see Figure 6.11). However, the overall best results are still provided by the basic method, as Figure 6.12 illustrates for the optimal term selection. A similar observation is made by Syed et al. who state that using spreading activation over category links "results in the same prediction or a prediction of a more generalized concept that is evident in the results" [SFJ07].

To conclude the results, the hierarchical propagation could not improve the categorization effectiveness compared to the basic method. Regarding $acc_{hier}$ the modified method is at best equal to the basic method, whereas it performs significantly worse regarding $acc_{flat}$.

### 6.3.3 Experiment 3 – Fragmentation of Learning Resources

A further idea for a modification of the categorization method is to take advantage of the internal structure of a learning resource. Textual learning resources are often organized internally as either a tree of screen pages or a sequence of screen pages. These pages are called fragments of a learning resources. Hearst observes for application of information retrieval methods that "a long text is often comprised of many different subtopics which may be related to one another and to the backdrop in many different ways" [Hea95]. We believe that this effect may affect not only information retrieval but also machine learning methods. If a document contains several subtopics, methods such as the k-nearest-neighbors algorithm could be impaired by term vectors of mixed-topic documents. Therefore, the idea is to apply

[39] diagrams for other term selections are similar

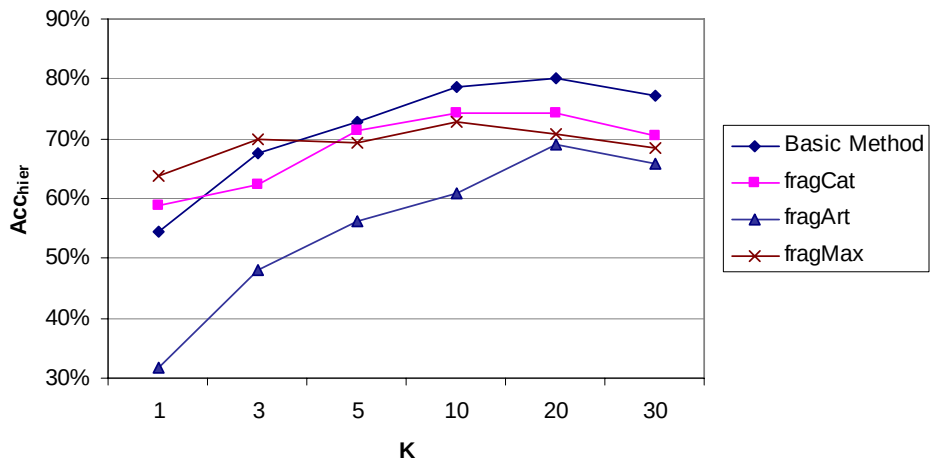


Figure 6.14: Comparison of fragment-based categorization (hierarchical accuracy).

the kNN method to each individual fragment instead of the whole learning resource in order to obtain subtopics of a learning resource. In a further step the overall topic has to be inferred from the sum of all subtopics. The fragmentation of learning resources has been derived from the corresponding SCORM manifests.

Three different approaches were tried of how to determine subtopics and afterwards the overall topic of a learning resource. All three variants are based again on the ranking of articles based on their similarity with a given term vector. The difference to the basic method is that term vectors are used per fragment instead of per learning resource. According to the calibration results of the first experiment only the cosine measure was used and a term selection of $3 \leq DF \leq 100,000$ was applied.

The first approach, called $\text{frag}_{cat}$, determines the most probable category for each fragment of a given learning resource, analog to the basic method. In fact, the basic method is directly applied to each fragment. In a second step, the occurrences of each category over all fragments are counted. The most common fragment category is adopted as the category of the whole learning resource (see Algorithm A.1). This method variant bases upon the assumption that subtopics of a learning resource belong to the same category.

A second approach, $\text{frag}_{art}$, determines the $k$ most similar articles for each fragment (see Algorithm A.2). But instead of inferring an individual category for each fragment as above, the determined articles from all fragments are merged into a large set (duplicates are allowed). This joint set of articles is used to determine the overall category again by counting the occurrences of each category.

The third approach is similar to the second one. Article similarities are calculated for each fragment. But in contrast to the first two approaches, those $k$ articles are selected for the overall learning resource, which have the maximum similarity values in any of the fragments. Therefore, the variant is named $\text{frag}_{max}$ (see Algorithm A.3). This method tries to benefit from fragments, which are focussed on particular subjects. Fragments that are less focussed on a particular topic are intended to have less impact on the categorization. For this method, the fragmentation is extended: While the former methods only considered leaf nodes of the internal tree structure of a learning resource, for $\text{frag}_{max}$ also non-leaf elements are included by the algorithm.

Table 6.2: Categorization performance for three fragment-based methods.

| method | measure | $k=1$ | $k=3$ | $k=5$ | $k=10$ | $k=20$ | $k=30$ |
|---|---|---|---|---|---|---|---|
| basic | $acc_{flat}$ | 30.00% | 43.00% | 51.00% | 59.00% | 62.00% | 58.00 |
| method | $acc_{hier}$ | 54.35% | 67.68% | 72.76% | 78.51% | 80.02% | 77.32% |
| cat | $acc_{flat}$ | 29.00% | 36.00% | 49.00% | 53.00% | 54.00% | 48.00% |
| | $acc_{hier}$ | 58.96% | 62.39% | 71.27% | 74.36% | 74.32% | 70.38% |
| fragAcc | $acc_{flat}$ | 13.00% | 20.00% | 27.00% | 32.00% | 49.00% | 43.00% |
| | $acc_{hier}$ | 31.61% | 47.93% | 56.11% | 60.84% | 69.15% | 65.77% |
| fragMax | $acc_{flat}$ | 38.00% | 47.00% | 45.00% | 52.00% | 49.00% | 46.00% |
| | $acc_{hier}$ | 63.83% | 69.89% | 69.34 % | 72.88% | 70.90% | 68.53% |

These three methods for fragment-based categorization were implemented and performed on the already known calibration set of 100 learning resources. The results are listed in Table 6.2. Note that the $k$ value has a slightly different meaning for these three methods. The Figures 6.13 and 6.14 visualize the accuracy curves of the three fragment-based categorization methods and the basic method. For low values of $k$ the $\text{frag}_{max}$ method performs better than the basic method. However, the overall best accuracy is still achieved by the basic method.

It is obvious that the fragment-based categorization did not improve the categorization performance; it even performed significantly worse. Where do these results come from? A closer look at individual learning resources reveals that some of the learning resources were in fact classified better with fragmentation. However, for most learning resources, fragmentation leads to misleading articles, which are associated with the fragments. Partially, this effect comes from particular genres of the fragments: some learning resources contain case studies of diseases. These case studies are often assigned to the categories *Mann* or *Frau* (the German equivalents to *Men* and *Women*) by the classifier. If the whole learning resource is represented by a single term vector, this genre effect has less impact on the determined category. A reason, why the genre effect occurs so often are the relatively low similarity values for a given learning resource or fragment vector, which typically range between 0.1 and 0.2. Thus, a genre-specific tendency towards higher frequencies for some terms may impact the similarity function enough to spoil the article order. The implemented fragment-based methods do not utilize the position of a fragment within the learning resource. A weighting of fragments according to their position might help to improve the effectiveness.

## 6.4 Evaluation of the Wikipedia-Based Classifier

The previous sections have described, implemented and calibrated a new categorization method for learning resources. This method is intended to work domain-independently, without an existing training set of learning resources. Instead of a traditional training set, articles of Wikipedia were used as training corpus. Section 6.3 applied the new method to a calibration set of 100 learning resources. A set of parameter values was determined under which the method provided optimal results with regard to the calibration set. Although the method successfully worked for that particular set of documents, this

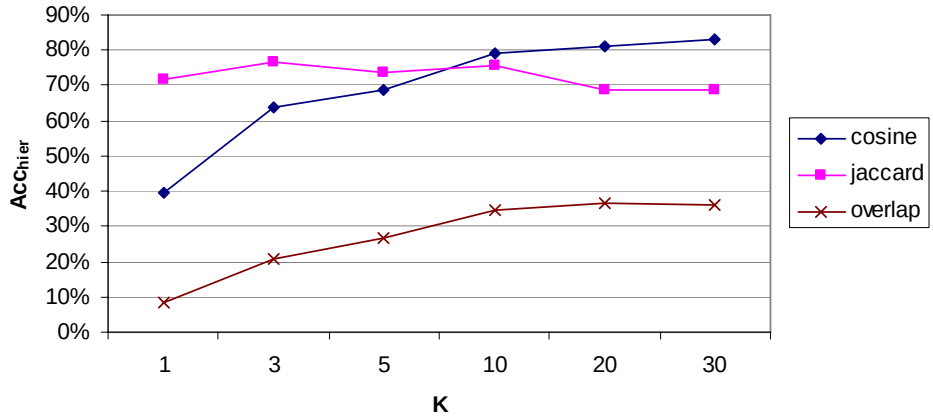


Figure 6.15: Hierarchical accuracy measurement for evaluation set (term selection: $3 \leq DF \leq 100,000$).

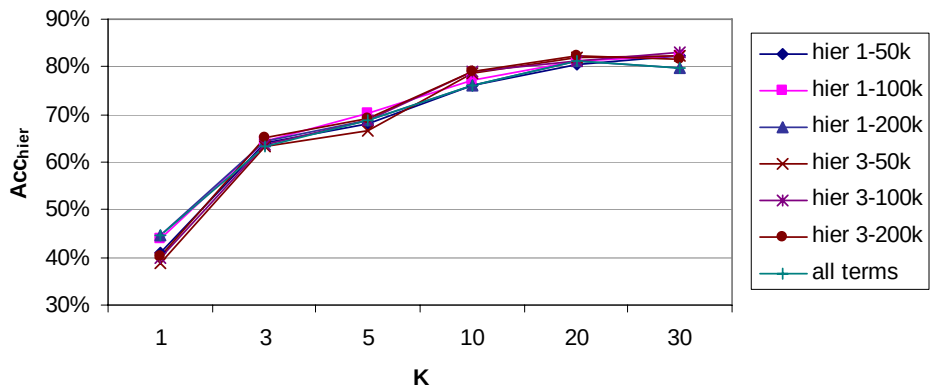


Figure 6.16: Comparison of different term selections with the cosine measure (hierarchical accuracy).

experiment is no proof that the method generally works, or that the determined parameters also provide reasonable results for other learning resources. If the determined optimal parameter settings produced bad results when applied to new learning resources, the determined parameters would have little value. Thus, it remains for this section to prove that the method as well as the determined parameters are stable (the method and the determined parameters provide similar results also for other learning resources). This proof is tackled in the first subsection. In addition, the quality of the Wikipedia-based classifier should be assessed by comparing it to existing categorization methods. This evaluation is covered by the second subsection.

### 6.4.1 Evaluation of Calibrated Parameters

The first part of the evaluation analyzes the stability of the new categorization method. Stability may refer to the overall quality of the method or to the behavior of the method regarding the free parameters. Regarding the stability of the overall method, the effectiveness of the method – when applied to another set of documents – should remain roughly the same. Parameter stability can be assumed if the same parameter values applied to two sets of documents provide comparable results (relative to the results of other parameter values).

For this evaluation a second set of documents is used. This set consists of another 50 learning resources from the k-MED project. As before, these learning resources were manually labeled. The classifier from Section 6.2 was applied to this set; the effectiveness was measured again as two separate accuracy measures (flat accuracy and hierarchical accuracy). In order to assess the overall stability, the achieved maximum accuracy values were compared to the results of the previous section. A maximum flat accuracy of 54% (calibration set: 62%) and a hierarchical accuracy of 82.9% (calibration set: 80.3%) could be reached. Hence, the flat accuracy was 13% below the first measurement, whereas the hierarchical accuracy was about 3% above the previous result. Particularly the more expressive hierarchical accuracy indicates that the overall performance of the method is indeed stable.

In a next step the stability of the parameters was considered. As Figure 6.15 shows, the three applied similarity measures provided results that are comparable to the calibration measurements. Again, the cosine measure performs best. Figure 6.16 provides a comparison of the available term selection sets over the different values of $k$. The curves look very similar to the calibration results. A difference can though be observed. There are now two optimal combinations of term selection and $k$:$(3 \leq DF \leq 100,000, k = 30)$ and $(3 \leq DF \leq 200,000, k = 20)$. The differences between $k = 20$ and $k = 30$ are though only marginal for both term selections. Consequently, the method can also be considered to be stable with respect to the choice of parameters.

### 6.4.2 Baseline Comparison

The experiments from the previous sections have shown that the Wikipedia-based categorization method works. But it is also desirable to know how it compares to existing methods. The performance measurement of an already existing categorization method on a given categorization task is called a baseline. The Wikipedia-based classifier has to be compared against the performance of the baseline method.

The novelty of the presented categorization method is the usage of Wikipedia articles as corpus documents for training a machine learning classifier. A baseline method would use a traditional corpus; in our case this traditional corpus would consist of real learning resources. The typical approach of creating a training corpus for categorization is to split the available documents into two sets, a training set and a test set. The training set is used as input for training a classifier; the evaluation set serves for measuring the categorization performance of the learnt classifier.

Based on these considerations, the available learning resources have been split into a training set of 100 learning resources and a test set of 50 learning resources. The learning resources belong to 63 different categories. The training set has been used for learning four different classifiers. The applied model learners are a Naive Bayes classifier (Bayes), a support vector machine (SVM), a J48 decision tree learner (J48), and a JRip rule learner (JRip). The Weka workbench[40] has been used as environment to train and evaluate these classifiers. As performance measure the flat accuracy is used. The classifiers have been trained with different feature options. As features a weighted (TF-IDF) term count was used. First all features were applied for training; afterwards, 100 features were selected using the $\chi^2$ method [CMS01]. Features were used as real and as binary values. For some of the classifiers the full feature set could not be used due to memory limitations.

[40] http://www.cs.waikato.ac.nz/ ml/index.html

Table 6.3: Comparison of Wikipedia-based classifier with baseline methods.

| Classifier | $Accuracy_{flat}$ (feature selection, real-valued) | $Accuracy_{flat}$ (feature selection, binary) | $Accuracy_{flat}$ (all features, real-valued) | $Accuracy_{flat}$ (all features, binary) |
|---|---|---|---|---|
| JRip | 4% | 0% | 0% | 0% |
| J48 | 4% | 2% | 2% | 2% |
| Bayes | 4% | 2% | – | – |
| SVM | 2% | 8% | – | – |
| J48 / AdaBoost | 4% | 2% | 4% | 2% |
| Bayes / AdaBoost | 6% | 2% | – | – |
| SVM / AdaBoost | 2% | 8% | – | – |
| Wikipedia-based | 52% | | | |

The classification results (Table 6.3) indicate a very poor effectiveness of the baseline methods. Accuracy values between 0% and 8% were achieved. The best conventional classifier was a support vector machine using selected binary features. The number of documents per class varies strongly, which may affect the performance. Therefore, three of the baseline methods (Bayes, J48 and SVM) were additionally combined with the boosting learning scheme AdaBoost.M1 [FS97]. AdaBoost repeatedly trains a classifier and adapts the weight of sample documents between the rounds in order to improve the effectiveness of weakly classified categories. However, the maximal accuracy of 8% could not be further increased.

The baseline comparison has shown that the traditional methods can learn only poor classifiers from the given training set for about 60 categories. The Wikipedia-based classifier[41] performed significantly better for these categories; it is assumed to provide a similar effectiveness for the remaining 40,000 categories.

Saini et al. report an effectiveness (F1 measure) of up to 69% for categorization of learning resources into 20 classes; this result has been achieved under the condition that there are about ten sample documents per class [SRS06]. For our scenario with a large number of categories (more than 40,000) it is unlikely that there will ever be a suitable training set of 400,000 manually labeled documents. Even for a fraction of that number (63 classes) the baseline comparison has demonstrated that a ratio of about 1.5 documents per class prevents a successful learning of classifiers. By contrast, the Wikipedia-based classifier is able to classify the same test set with a much higher accuracy. Thus, the Wikipedia-based classifier outperforms conventional methods under the conditions of the underlying scenario.

## 6.5 A New Approach for Generation of Pedagogical Metadata

The previous sections have dealt with a new method for generating topical metadata. However, learning object metadata contains more fields besides topical information. For instance, several metadata fields

[41] For this measurement the determined parameters from the calibration experiment were used, leading to 52% accuracy. Up to 54% accuracy could be reached for the test set with different parameters.

for pedagogical purposes are defined [Hod02b]. This section briefly presents an approach for generating pedagogical metadata. The method classifies learning resources of a particular granularity regarding their pedagogical function.

### 6.5.1 Categorization of Information Objects Using Textual and Structural Features

Learning resources are commonly hierarchically structured, consisting of multiple levels of granularity. Hence there is a chance that some parts of a course at a finer granularity could also be re-used in other courses. The granularity level that is most promising for re-use is the level of so called information objects. Information objects are parts of a learning resource (typically one or a few screen pages) that each have a dedicated didactic function, such as an overview, a theorem, an example or a test. A large number of possible didactic functions are described by Meder's didactic ontologies [Med00]. Multiple information objects are aggregated to form a learning object, which is suited to achieve a particular educational objective.

Having pedagogical metadata available could improve retrieval of learning resources beyond mere topical queries. Hence it would be helpful if each information object were labeled with its didactic function type. Unfortunately, authors tend to maintain metadata very sparsely; didactic function types are rarely available. Automatic metadata generation is an umbrella term for different methods to automatically create missing metadata, e.g. by means of machine learning technology. Up to now, classification of didactic functions has not been addressed by any existing metadata generation method. This subsection presents our recent work concerning automatic classification of the didactic functions of information objects. Full details of the experiment setup and achieved results can be found in [Han07].

### 6.5.2 Textual and Structural Features for Pedagogical Classification

Machine learning methods have already been discussed in Section 6.1. Classifiers do not take complete objects as input but require mapping the objects to a set of features. Typical features of text documents, for example, could be occurring words, but other attributes, such as document size or average length of sentences are also imaginable. In the case of multimedia content (e. g. images or videos), more sophisticated features are needed – for instance color histograms, thickness of lines or detected objects. Many classification systems for text-based documents rely solely on textual features. Textual features can be divided into simple statistical information (such as word occurrences) and natural language analysis. Examples of the latter would be lexical chains [MH91], word sense disambiguation [San94] or grammatical mood.

For the approach of Section 6.2 only occurring words from a document were used as features. This works fine if a topical classification is intended. But classification of information objects by didactic functions is comparable to the task of genre detection [SFK00] – both classify not the subject but rather another dimension of the document. Hence, using only textual features overrates the subject dimension of a document. For didactic classification additional features are potentially useful. Besides textual features, Web based trainings also contain multimedia aspects, which are likely to differ between different didactic functions. For instance, the presence of interactive media, such as flash animations or usage of

Table 6.4: Selected features for classification.

| Feature name | Description |
|---|---|
| WORD_COUNTER | Length of the text |
| JS_COUNTER | Number of JavaScript functions |
| CONTAINS_LIST | HTML code contains at least one list |
| CONTAINS_FORM | HTML code contains forms |
| CONTAINS_INPUT | HTML code contains input elements |
| CONTAINS_CHOICE | HTML code contains choice elements |
| CONTAINS_INT | HTML code contains interaction elements |
| CONTAINS_SWF | Flash animations are embedded |
| HEADLINE_KW | Significant keywords that have been found in the page headline |

scripting languages (e. g. JavaScript) could be an indicator for assessments or demonstrations, whereas they are less likely to appear in other information object types.

Several possible features have been specified for the intended classification task. They have been categorized into linguistic features, recurring structural patterns and hypertext features. Linguistic features could be the total text length, occurrence of key terms, headlines and sentence types. Recurring structural patterns are the position of an information object within the tree structure of a SCORM package or special knowledge about patterns in other courses from the same author or authoring tool. Hypertext features are structural similarity of HTML documents, referenced style sheets, in-link analysis and embedding of interactive media contents or scripts.

### 6.5.3 Experimental Setup for Pedagogical Classification

An experiment was set up to evaluate whether multimedia features can be used for classification of didactic functions. Learning resources from two sources have been used. One source was again the k-MED project that has been already described above. Other samples were taken from the Content Sharing project[42]. Thus, the samples have been created from two different authoring environments and by multiple authors. The sample courses have been split into information objects. In total, 166 information objects were used for training and 207 samples for performance measurement.

Each information object was manually labeled with its didactic function. The available didactic function types were taken from Meder's didactic ontologies [Med00]. According to the didactic ontologies, the function types are hierarchically ordered on three levels of detail. The first level of detail differentiates between receptive knowledge types and interactive assessments. Receptive knowledge types are further subdivided into source knowledge, orientation knowledge (facts or overview), explanation knowledge (what-explanation or example) and action knowledge (checklist or principle). Interactive assessments are either multiple choice tests or assignment tests.

For the implementation of the classification task the free classification framework Weka was used. Four different classifiers were evaluated: a Bayes network classifier (Bayes), a support vector machine

[42] http://www.contentsharing.com

(SVM), a rule based learner (JRip) and a decision tree learner (C4.5). Human judgment was chosen as the baseline for comparing the automatic classifiers against. Six people were asked to manually classify the given samples.

Nine different features were selected for the experiment. These features take into account not the pure text but rather multimedia aspects. Furthermore, most of the features are independent of the particular course language. The only textual feature is the headline keyword class. For this feature, particular decisive words that may occur in a headline are mapped to one of a set of keyword classes. The features are listed in Table 6.4.

The didactic ontology has three levels of detail. The performance of a classifier can be measured for each of the levels. Furthermore, it is also possible to classify hierarchically. A first classifier decides only which top-level category an information object belongs to, the second classifier decides at the middle level and a third classifier categorizes only on the highest level of detail. Each classifier uses the result of the previous classifier as additional feature of the object being classified. The experimental setup was arranged to allow both flat and hierarchical classification.

### 6.5.4 Evaluation of Pedagogical Classification

As described in the previous subsection the chosen classifiers were trained with a training corpus of 166 information objects. 207 further information objects have been available as test corpus for classification performance evaluation. Most of the experiments were set up as single-label classification; that is, each information object is assigned to exactly one category. Six people were asked for their judgment in order to obtain a baseline. The classifiers are evaluated on three levels of detail according to the three levels from Meder's didactic ontologies. As in the previous sections *accuracy* was chosen as effectiveness measure. Accuracy is calculated here as

$$Accuracy = \frac{|correctly\ classified\ samples|}{|total\ number\ of\ samples|} \tag{6.13}$$

The first experiment was classification at the lowest level of detail; that means the classifiers must decide only if a given information object is either a knowledge type or an assessment. The experiment was performed first with all nine features and afterwards with a selection of six features: JS_COUNTER, CONTAINS_FORM, CONTAINS_INPUT, CONTAINS_INT, CONTAINS_SWF and HEADLINE_KW. All four classifiers performed the task with an accuracy of 100% after feature selection. Without feature selection, the Bayes and JRip classifiers achieved only 99%. The result became clear after a closer look at the information objects: all assessments contain markup elements that enable user interaction, whereas most knowledge types do not have these elements. Thus, the markup-based features chosen are very decisive for distinguishing between knowledge types and assessments.

The next experiment was classification at the second level of detail. On this level, there are two different types of assessments and three different knowledge types. These five level-two types are used as categories. It is assumed that no information about the lowest level of detail is known. First, the clas-

Table 6.5: Classification of second level of detail.

| | Bayes | SVM | JRip | C4.5 | Human Baseline |
|---|---|---|---|---|---|
| Accuracy (all features) | 0.579 | 0.613 | 0.585 | 0.618 | 0.787 |
| Accuracy (selected features) | 0.609 | 0.613 | 0.604 | 0.614 | |

Table 6.6: Flat classification of highest level of detail.

| | Bayes | SVM | JRip | C4.5 | Human Baseline |
|---|---|---|---|---|---|
| Accuracy (all features) | 0.396 | 0.382 | 0.367 | 0.430 | 0.618 |
| Accuracy (selected features) | 0.391 | 0.381 | 0.353 | 0.454 | |

sifiers were trained with all nine features. In a second run, only three selected features[43] were used as input: CONTAINS_FORM, CONTAINS_CHOICE, HEADLINE_KW. The evaluation results are compared in Table 6.5.

First of all, the performance of human judgment is noteworthy. Apparently, the sample information objects could not be assigned as clearly to one of the knowledge types as theory suggests. Human judgment achieved an accuracy of 78%. This value also has another implication: Retrieval systems should consider that different users, who are looking for the same information object, may search by different attribute values. The C4.5 classifier showed the best performance compared to the other classifiers, both with all features and with a reduced feature set. The Bayes and JRip classifiers improved by reducing the number of features, whereas the C4.5 classifier slightly degraded. The differences between the four classifiers shrunk after feature selection. The results of the feature selection indicate that the two types of assessments can be distinguished by different types of interactive HTML markup. But markup is not significant for differentiating different knowledge types; of all examined features headline keywords are most expressive. Future experiments should find out whether linguistic features may result in a better performance.

The next experiment evaluated the performance of classification for the highest level of detail. This level of detail consists of eight classes. A flat classification is assumed; this means that no classification information from the other levels of detail is known. The performance results are denoted in Table 6.6. These results are less accurate than those of the second level of detail. Even the best classifier C4.5 has achieved only 43% using all features, which is almost 20% below the human baseline. Feature selection slightly improves the performance of the tree learner: using only the features WORD_COUNTER, CONTAINS_FORM, CONTAINS_CHOICE and HEADLINE_KW raises the accuracy value to 45%. This accuracy is still too low for practical application and needs to be improved.

Table 6.7: Hierarchical classification of highest level of detail.

| | Bayes | SVM | JRip | C4.5 |
|---|---|---|---|---|
| Accuracy (all features) | 0.700 | 0.680 | 0.667 | 0.660 |
| Accuracy (selected features) | 0.705 | 0.686 | 0.686 | 0.676 |

[43] Features have been selected using the information gain measure of Weka.

An approach for increasing the performance is hierarchical classification, which means that there is a separate classifier for each level of detail. Each classifier uses the category information from the lower level of detail as additional feature. Hierarchical classification was tested at the highest level of detail having the category from the second level available as known input. First, all features were used. Afterwards, only the features WORD_COUNTER, CONTAINS_LIST, HEADLINE_KW and CLASS2 (known second-level category) were selected for determining the third level of detail. This time, the Bayes network showed the best performance both for all features and for just the selected features. In both cases an accuracy of 70% has been reached.

## 6.6 Summary

In this chapter the automatic generation of metadata has been covered. The availability of metadata is a key to successful retrieval of learning resources. Two contributions to this area of research were made. The first contribution is the development of a new, Wikipedia-based method for topical classification of learning resources; and second, the classification of pedagogical types of learning resources based on textual and structural features was analyzed.

The motivation for developing a new classification method was the observation that the scenario of a typical domain independent and in the beginning empty repository poses a challenge that existing methods cannot cope with. Such a learning object repository is considered to contain various learning resources of very different subject areas. Conventional classification methods require a training set of documents for learning how to classify correctly. For successful learning, a ratio of ten or more sample documents for each individual category is required. In the present scenario the number of categories is assumed to be large, because the diversity of subject areas has to be regarded, as well as the differentiation of topics within a subject area. Unfortunately, suitable training sets of several thousands of manually labeled learning resources are hard to obtain.

Therefore, a Wikipedia-based classifier has been proposed as a new approach that works without a conventional training set. Instead of manually labeled learning resources of the type typically found in repositories, encyclopedic articles are utilized as training documents for learning a classifier. Wikipedia, a well-known online encyclopedia was chosen as training corpus. The k-nearest-neighbors method was chosen as underlying classifier approach. A review of related work has shown that Wikipedia articles have not been used as training documents, before, neither for the k-nearest-neighbor method, nor for other machine learning methods. This chapter has introduced a novel Wikipedia-based classifier and has provided implementation details. The new method has been evaluated in two ways. First, the impact of some parameters and method variations on the effectiveness of the method were analyzed. Afterwards, the validity of the method was evaluated by checking the consistency of effectiveness measurements between two independent document sets. Additionally, the measurements were compared to some baseline methods.

The analysis of parameters and method variants showed that the choice of a term vector similarity measure and the specification of the $k$ parameter had the most impact on the result quality. The cosine measure proved to outperform other similarity measures, a $k$ value of 20 produced optimal results. Term selection has a positive effect, but to a much lesser degree than the other two parameters. Two additional

variants of the Wikipedia-based classifier – hierarchical propagation and fragmentation – have been tried. However, these modifications cannot keep up with a reasonable optimization of the basic parameters. It can be concluded for the two tested variants that complex modifications of the Wikipedia-based classifier do not increase the classification accuracy.

An evaluation of the classification method by comparison of accuracy measurements on two separate document sets was performed. The results for both sets were similar. Particular the determined optimal parameter values of the calibration set achieved almost the best results with the validation set (see Section 6.4). Furthermore, a comparison of the Wikipedia-based classifier with conventional classification method proved that the Wikipedia-based classifier performs significantly better than the baseline methods (54% accuracy vs. 8% accuracy). This test also highlighted that the baseline methods are not applicable in the present scenario.

As the evaluation has shown, the Wikipedia-based classifier works well. There is still potential for further improvements and for extended use cases. For instance, the cross references between Wikipedia articles could be leveraged. There are often references from one article to other articles, which are related to the contents of the first article. This relatedness of articles might be taken into account as additional information for improving the classification.

This chapter has also made a contribution towards didactic classification of information objects using machine learning technologies. First, different types of features, which might be relevant for this task, have been identified. Then a series of experiments have been presented, where, in particular, multimedia features (such as markup or embedded interactive contents) have been used for automatic classification. According to the chosen categories from a didactic ontology, the classification performance has been evaluated at different levels of detail. The coarsest level of detail only differentiates between receptive knowledge types and interactive assessments. At this level a classification accuracy of 100% could be achieved. However, this performance was partially due to the special characteristics of the evaluated data sets. On finer levels of detail the accuracy decreased. On the second level of detail the accuracy amounted to about 61%. On the third and finest level of detail only 45% accuracy could be achieved. This value could be raised to 70% by applying hierarchical classification.

These experiments have indicated that automatic didactic classification of information objects is possible and that multimedia features – especially markup information – are suitable features. However, the performance requires significant improvement for use in practical applications. This might be achieved by taking into account further types of features. Two types of features, which were not used in the discussed experiments, appear to be especially promising. The first additional feature type is the position of an information object within the learning object or course it belongs to. An argument for using the position as a feature is that the arrangement of information objects is often influenced by an author's intended learning strategy. The second promising feature type is linguistic information; the style of speech of didactic texts varies depending on the particular didactic intention. Thus, linguistic features may complement the feature set to achieve a higher performance.

# Part IV

# Proof of Concept and Conclusions

# 7 Prototypical Implementation

As Chapter 2 has outlined, multi-granularity reuse of learning resources in heterogeneous systems is the overall goal of this thesis. Chapters 3, 4 and 5 have developed a consistent set of concepts that aim at improving different constituting features of multi-granularity reuse. There are particularly concepts for modularization, aggregation, improved retrieval of learning resources, and a content representation concept for a re-purposing framework.

The concepts have been evaluated by implementation in a prototypical application, the so called Re-purposing Tool Suite. The environment for this implementation has been the Content Sharing project, which is described in Section 7.1. Section 7.2 presents the overall Re-purposing Tool Suite that serves as a platform for different re-purposing tools, such as a modularization tool, an aggregation tool and an adaptation tool. The tool suite also provides support for the SCORM-based module concept from Section 3.1.2. Three particular re-purposing tools, which have been realized within the Re-purposing Tool Suite, are described in Section 7.3. At the end of this chapter, experiences from the deployment of the Re-purposing Tool Suite in the Content Sharing project are discussed.

## 7.1 The Content Sharing Scenario

The Content Sharing project was a public funded research project with several German companies and educational institutions as partners [The]. The overall goal of the project was to establish an online marketplace for learning resources, especially for a low price market segment. Intended user groups for the market place were especially small content producing companies, small and medium-sized companies with demand for cost-efficient e-learning, educational institutions, and teachers for vocational training. It is assumed that the best way to achieve cost-efficient access to e-learning contents is to enable modular exchange and re-purposing of contents between users of the Content Sharing system.

The Content Sharing project has addressed the issue of learning resource exchange and reuse from different perspectives, in particular from organizational, economical, legal and technical perspectives. For the rest of this chapter, only the technical aspects of the Content Sharing project are of interest.

From a technical point of view, the Content Sharing project has developed two components: a learning object repository and a local Re-purposing Tool Suite that enables users to modularize, aggregate and adapt learning resources, which they have either obtained from the marketplace, or want to offer via the marketplace. Both, the repository and the local Re-purposing Tool Suite support modular learning resources; but it is also required, that users can directly reuse obtained learning resources without using the Re-purposing Tool Suite.

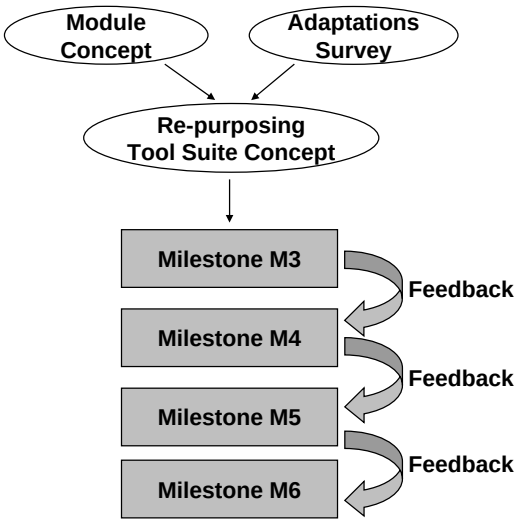


Figure 7.1: Implementation milestones of the Re-purposing Tool Suite.

### 7.1.1 Re-Purposing of Learning Resources

The Content Sharing scenario assumes that the learning object repository is used by a heterogeneous group of content producers, teachers and learners. These user groups use different authoring tools and learning managements systems. In order to be economically successful, the Content Sharing platform should not establish high technological barriers, which prevent users from using the platform. The project has decided to recommend SCORM as the preferred format for learning resources. The concept of modular SCORM learning resources (also simply called *modules*) presented in Chapter 3 has been implemented as the basis for modular learning resources. A transformation between conventional SCORM packages and modular SCORM learning resources is automatically performed by the tool suite.

The tool suite supports re-purposing processes, which have been demanded by potential users. A survey has revealed which types of adaptation are most relevant to the users [ZBRS06]. Beside these adaptation types, modularization and aggregation are supported by the Re-purposing Tool Suite. Intellectual property issues of reuse of learning resources have not been regarded. Digital watermarking [Dit00] could be used to track the reuse of contents.

### 7.1.2 Iterative Development Process

The components of the Content Sharing system – the central repository and the local Re-purposing Tool Suite – have been developed iteratively. The concepts and prototypes have been presented to potential users several times to get feedback as early as possible. The concepts have been adapted according to the user feedback; features have been added, changed and removed.

In total, the development of the Content Sharing system was organized in six milestones. Of these six milestones, the last four included technical prototypes of the Re-purposing Tool Suite (see Figure 7.1). In the beginning, a module concept (see Section 3.1.2) has been developed. This concept serves as the basis for the LOR and the tool suite. In parallel, a survey was performed to find out, which adaptations potential users would like to perform, and how [ZBRS06]. These two preliminary works resulted in an

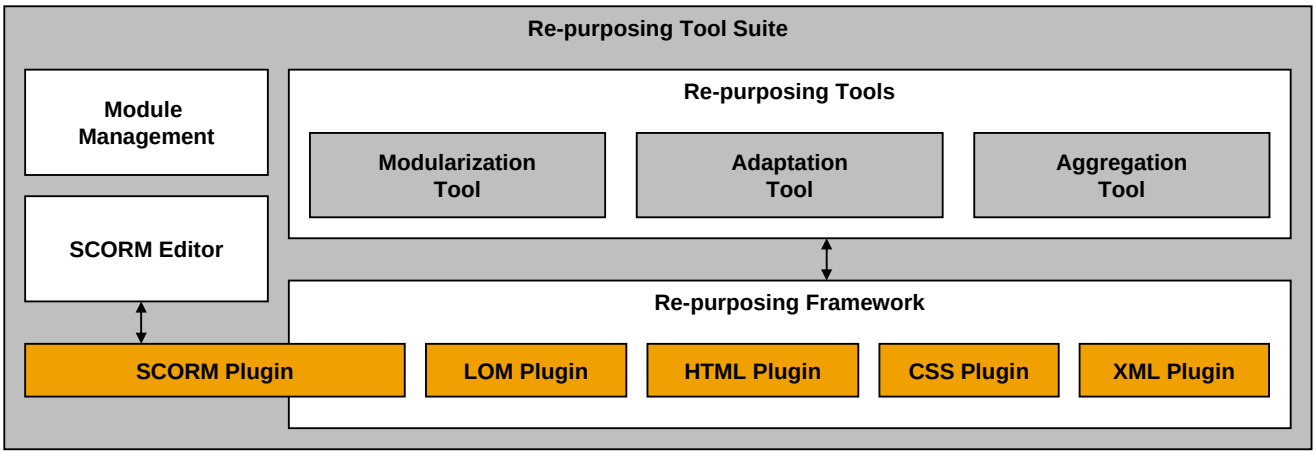


Figure 7.2: Components of the Re-purposing Tool Suite.

overall concept for the Re-purposing Tool Suite as a framework and for the individual tools. The resulting tool suite and the contained tools are presented in the following sections.

## 7.2 A Re-Purposing Tool Suite

This section describes the overall Re-purposing Tool Suite; the included re-purposing tools are separately described in the succeeding section. Based on the scenario of the Content Sharing project functional requirements for re-purposing tools were analyzed. These functional requirements led to a set of components for a Re-purposing Tool Suite. Besides components for modular operations (modularization, aggregation, and adaptation) some more components turned out to be necessary. For instance, basic editing functionality for SCORM structures or a component for module management. Authors sometimes require an editor to perform minor changes. Discussions with users have shown their desire to view the structure of a learning resource and a preview of the contents. Without an overview of the course contents, the users do not get a sense of what they are working with. Furthermore, some changes of a learning resource do not require complex re-purposing tools. For instance, a user might want to change the order of some sections, or edit some titles. For these changes, a simple SCORM editor is the tool of choice.

The tool suite comprises several components as Figure 7.2 depicts. One component is a workspace and module manager, which handles the management, import, export and revision control of modules. An integrated SCORM editor is the second component; it allows users to perform simple changes of the course structure of a module. Besides these two components, a re-purposing framework implements the abstract content representation model proposed in Chapter 4. An application programming interface allows to plug different re-purposing tools for modularization, aggregation and adaptation into the tool suite and operate upon this content representation. These components will be presented in the following subsections. The whole tool suite has been developed in Java 1.5, using the Eclipse Rich Client Platform for graphical user interfaces and plug-in mechanisms, which are introduced later in this chapter.

### 7.2.1 Implementation of Modular Learning Resources

This section describes how the concept of modular learning resources has been realized in the Content Sharing system. The Content Sharing project basically uses the modular SCORM concept as specified in Chapter 3 of this thesis. A module is a SCORM-based learning resource, which complies to certain additional requirements. The requirements identified in Chapter 3 are

- SCORM compliance
- Modularity requirements
- Support for metadata
- Enable modularity-awareness of repositories
- Support for modular operations
- Support versions and updates

From these general requirements, some implications for the specification of modular learning resources arise. Firstly, a specification has to be defined that enables compliance with SCORM and LOM. Another important issue is the support of modular aggregation and modularity-awareness. The specification has to ensure that aggregating the modules is supported; repositories and other systems should be able to deal with aggregations. Another requirement is support for revision control: particularly the combination of several modules (of which different versions might exist) is a challenge. The implementation of the tool suite has to deal with two additional issues: How are learning resources represented while an editing process takes place? And finally, there have to be transformations that allow an exchange of SCORM compliant learning resources between the tool suite and external systems.

#### Specification of Modular Learning Resources

The specification for modules has been defined as an extension of the SCORM specification. This facilitates to achieve SCORM compliance. Three requirements were crucial for the specification: modularity, support for aggregation (by reference), and support of several versions.

A prerequisite for modular aggregation is that each module can be *uniquely identified*. Furthermore, to enable the aggregation of modules, each module has a unique folder namespace. Diverging from the SCORM specification, all files of a module (including the manifest document) reside within the unique module folder. The Universally Unique Identifier (UUID) specification is used to create module identifiers [LMS05]. A UUID can be created by distributed systems and are yet assumed to be unique. New modules are created not only by the central repository, but also by local tools. Thus, it is essential that local applications can generate new unique module identifiers.

Modularity also implies that *revision control* has to cope with multiple versions of one module. It is likely that sometimes modules are revised which are already part of an aggregation. New versions of a module are created by copying and assigning a new unique identifier. Two newly introduced LOM relation types indicate that a module is either a revision or a variant of a referenced module. Revisions

and variants are the two types of versions, which occur in the Content Sharing scenario. A revision is a consecutive version (e. g. an update or correction) of an existing module. The revision is intended to replace the former version. A variant is an alternative version of a module, for instance a translation into another language, or an adapted version using another visual style.

### Aggregation of Modules

Aggregation is realized on three levels. On a metadata level, aggregation is expressed as a relation (LOM category 7) in the metadata record of the aggregating module. The relation contains the identifier of the included module. The vocabulary of relation types has been extended with the additional value *include_module*. It is further assumed, that all modules of a composed learning resource are available in the file system within a common root folder. The second level of aggregation uses the *xinclude*[44] mechanism for XML to link one manifest document into another one. The manifest of an included module is referenced by the aggregating module. Elements of the included module manifest are then available within the aggregating manifest. The third level of module aggregation is implemented by enabling references within the aggregating manifest to elements of an included manifest.

The aggregation of modules has an impact on SCORM compliance. How compliance with the SCORM specification can be ensured for aggregations will be dealt with below.

### Workspace Concept: Modules in Progress

The packaged form of a SCORM learning resource is rather unsuited to process the contents. In order to enable editing of contained documents of a learning resource, the archive file is unpacked into a folder of the the local file system. Two kinds of module storage are distinguished within the tool suite: workspaces and module pools. A workspace is a temporal module storage, which is the location for module creation and editing. A module pool is a permanent storage, in which finalized modules are kept.

The module pool contains modules, which are either intended to be edited, adapted or aggregated by the tool suite, or have been finalized after such a processing. Modules in the module pool exist as package archives – each module is located in a separate archive. Because versions of a module are considered to be different modules (they have different identifiers), it is possible to store multiple versions of a module in the same module pool.

The workspace holds unfinished modules, which are currently processed by the Re-purposing Tool Suite. All modules, which are loaded into the workspace, are unpacked into a shared workspace folder. As each module has a unique folder namespace, conflicts because of duplicate files or folders cannot occur. Whenever an imported module is changed for the first time, the module management component automatically assigns a new module identifier to this module and marks it as a version of the original module.

[44] http://www.w3.org/TR/xinclude/

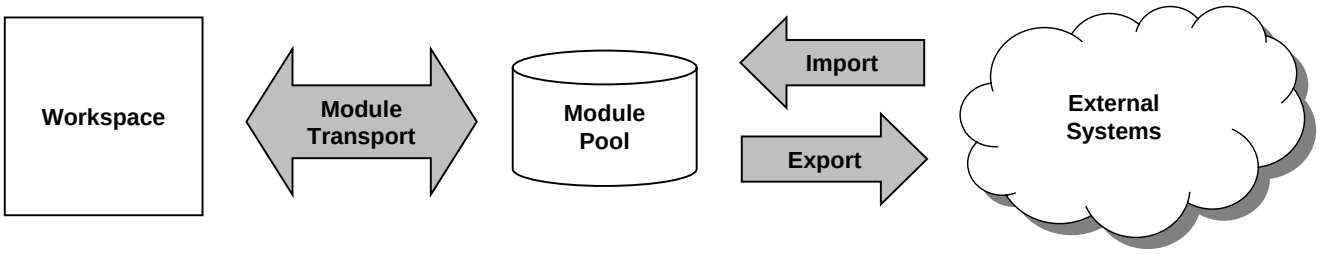


Figure 7.3: Import and export of modules.

## Import and Export of Modules

According to the scenario, existing SCORM compliant learning resources are intended to be reused. Learning resources that have been processed in the Re-purposing Tool Suite will be used afterwards in different systems (particularly in learning management systems). Therefore, learning resources have to be transformed between the original SCORM specification and the extended SCORM specification. This transformation is either an import of learning resources into the Re-purposing Tool Suite or an export of learning resources to external systems.

In total, there are four types of module transports between storage locations, which occur in the Re-purposing Tool Suite. These four transport types are the import of modules from an outside location into the module pool, the export of modules from the module pool to an external location, the transfer of modules from the module pool into the workspace and the transfer of modules from the workspace into the module pool. These types of module transports are illustrated in Figure 7.3.

Module transports between module pool and workspace are basically packing and unpacking operations. Aggregation relations from the LOM record of modules are read and all included modules are, too, transferred in order to keep the composed module complete.

The import and export of modules is more complex. The transfer of modules between the module pool and external applications cannot be based on separate archive files for each module. In order to preserve compatibility with the SCORM specification, interchange packages have to contain multiple modules. An interchange package may contain either one or multiple modules, and in addition to the module manifests also a compatibility manifest. The compatibility manifest specifies the whole structure of the composed learning resource. It is created by resolving the xinclude tags of module manifests and copying all module manifests into a single document. Nonetheless, an interchange package still contains separate modules, which are delimited by their module path within the interchange package archive. Yet, each module has its own module manifest.

The export of modules is similar to the transfer of modules into the workspace: a selected module and all included modules are copied into a temporal folder. Afterwards the compatibility manifest is created as explained above. Finally, the temporal folder containing all required modules and the compatibility manifest is packed into a zip archive. This archive, the interchange package, is a valid SCORM package.

Import of modules has to be distinguished by the type of input. Packages to import can be either modular interchange packages, or conventional SCORM packages. Interchange packages, which already contain modules, can be easily unpacked. The package is split into the contained modules, each module

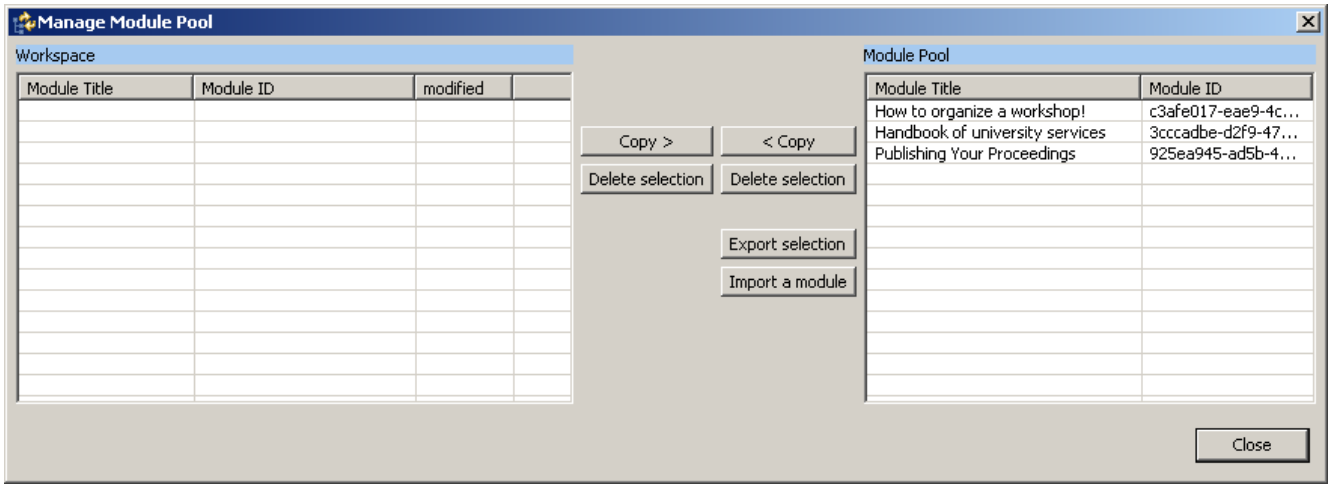


Figure 7.4: Screenshot of the module manager.

is stored as a separate archive in the module pool. The compatibility manifest is discarded. To import a conventional SCORM package, it has to be converted into a valid module. First, the package contents are moved from the root folder of the package into a new sub folder. Then, the metadata record of the manifest is completed by assigning a new module identifier. After these steps, the SCORM package has become a valid module and can be stored in the module pool.

### 7.2.2 Re-Purposing Framework

The basis of the Re-purposing Tool Suite is a re-purposing framework that implements the content representation model from Chapter 5. Three layers of content representations preprocess and present the contents of a learning resource to the integrated re-purposing tools. The overall architecture of the framework and its components are illustrated in Figure 7.5. The content representation consists of a physical representation (the documents of a learning resource), an Object-Oriented Content Representation, and a Semantic Content Representation. The last two content representations and how they are generated is described below.

The framework is able to handle multiple documents formats. Flexibility and extensibility has been achieved by providing an application programming interface (API) by which support components for different document formats can be plugged in. These components will be called document format plug-ins for the rest of this chapter. A document format plug-in provides all necessary classes for the Object-Oriented Content Representation and code to generate the OOCR out of the physical documents and the SCR out of the OOCR. Another API is provided for re-purposing tools to access the content representation and submit modifications to the framework. Document format plug-ins and the re-purposing tool API are presented in more detail below. According to the concept from Chapter 5 it is also envisioned that future components can generate semantic annotations of the content representation separate from the particular re-purposing tools. Components, which perform such an annotation are called Semantic Enrichment Components (SEC). Finally, the Modification Transaction Engine (MTE) is a component, which is responsible for receiving modification commands and controlling their execution.

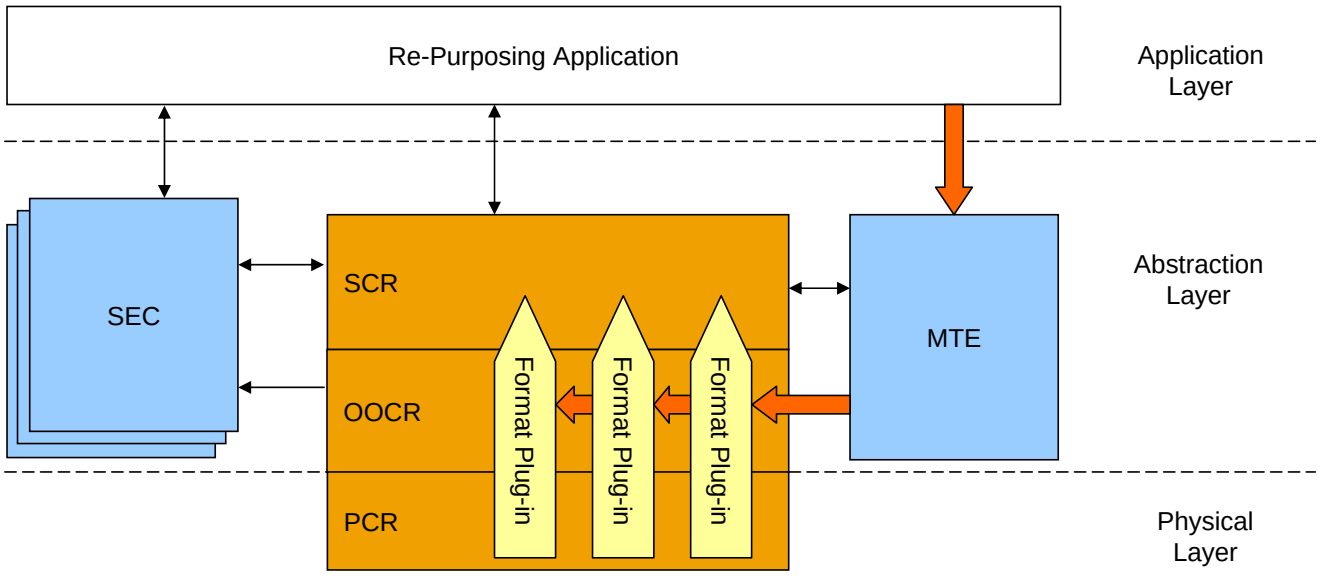


Figure 7.5: Architecture of the re-purposing framework.

### Object-Oriented Content Representation

The Object-Oriented Content Representation (OOCR) is a tree-structure of Java objects. Sub-trees of the overall tree represent the documents, which constitute the learning resource. There is a set of generic Java interfaces that specify the properties and methods of structural elements of the OOCR tree. Each document format plug-in implements these interfaces in format-specific classes. The archetypes for the OOCR have been document object models for XML and HTML documents, which provide an API for accessing, navigating and modifying XML and HTML documents in an object-oriented way [45]. Consequently, the document format plug-ins for SCORM manifests and HTML documents are internally built upon existing DOM libraries for XML and HTML. The generic interfaces particularly specify methods that can be used to traverse the OOCR tree for transforming it into the Semantic Content Representation.

Note that the reference structure of Java objects of the OOCR is not always a strict tree. It is even possible that a cyclic graph is produced, for example if two HTML documents exist, which mutually reference each other. The OOCR is insofar regarded as a tree, anyway, as the provided traversal methods avoid cycles and navigate over a logical tree. Some link relations have to be ignored during traversal the ensure the tree-property of the OOCR. However, these relations still exist and can be fully utilized after the transformation into the SCR.

### Semantic Content Representation

In order to perform high-level re-purposing tasks, such as adaptation of learning resources, abstraction may help to decrease the complexity of tools. As the OOCR is still very close to the original documents, another layer of abstraction has been introduced: The Semantic Content Representation (SCR) is an

[45] http://www.w3.org/TR/DOM-Level-2-Core/

abstraction layer that provides a view on a learning resource independent of the particular document format.

The OOCR is a content representation, which is based on Java objects. It is well suited for actively working with and modifying these objects. An analysis of the contents is only possible by navigating over the OOCR tree and performing checks during the traversal. Although all analysis tasks can principally be performed on this basis, this is not the most convenient one from a developer's point of view. Therefore, a second content representation layer has been introduced, the so called Semantic Content Representation (SCR). The SCR is an RDF[46] graph, which mirrors the structure and some properties of the content elements of the OOCR, but is optimized for other kinds of tasks than the OOCR. The elements of the Semantic Content Representation follow a common set of types, which stem from an underlying Content Ontology. This Content Ontology is specified in OWL[47], an ontology extension for RDF. More information about the Content Ontology used in the Content Sharing project can be found in [BFRS06]. The SCR may contain additional semantic information – properties and relations – that describe the elements in more detail. For instance, the links between HTML documents have already been mentioned. Another example is that the language of a text fragment can be provided as information, after it has been detected by a Semantic Enrichment Component.

In contrast to Java object graphs, RDF graphs can be directly queried for information by applications. There are several free RDF libraries available, for instance Jena[48] or Sesame [49]. Several query languages for RDF graphs have been proposed and implemented by the open source community [HBEV04]. For the present implementation, the Jena library has been used to build the SCR graph. The RDF query language RDQL, which has a SQL-like syntax, is used for some of the analysis tasks by the re-purposing tools. Besides query languages, the Jena framework also provides an API for navigating over the graph.

The Semantic Content Representation is generated after the OOCR has been completely created. The OOCR is completely traversed as a tree. For each element of the tree a node in the SCR graph is created. Afterwards, properties and relations for the SCR are inserted into the SCR. Which properties and relations are created and how this is actually performed depends on the particular document format and is thus the responsibility of the document format plug-ins.

There is one special feature of the SCR generation that has to be mentioned here. As it is intended to provide the SCR to re-purposing tools for analysis but perform modifications finally in the OOCR, it is essential that the framework is able to correctly associate nodes from the SCR with the corresponding OOCR objects. Therefore, this mapping is stored by the framework during the SCR creation. The association of an SCR node identifier with the corresponding OOCR Java object is written to a hash table, which can be used later to process modification commands.

### Semantic Enrichment Components

Users often demand more information about the contents of a learning resource than is originally available. Therefore, it is necessary to perform analysis methods on the contents to gain that information.

[46] http://www.w3.org/TR/rdf-primer/
[47] OWL Web Ontology Language, http://www.w3.org/TR/owl-features/
[48] http://jena.sourceforge.net/
[49] http://www.openrdf.org/

Integrating content analysis methods directly into the re-purposing tools may increase the complexity of these tools. Also, this approach would require that the same analysis method had to be implemented again by each tool that uses the method. In order to prevent redundant implementations, analysis methods can be outsourced to dedicated analysis components – the Semantic Enrichment Components. These components perform analysis tasks and add the gained knowledge to the SCR.

Two particular Semantic Enrichment Components, which have been developed for the Re-purposing Tool Suite, should be mentioned here. The first one recognizes the language of text fragments and writes this information as annotation to the SCR. The language recognition is used as a preprocessing step for a translation tool. A second SEC detects particular didactic functions of text fragments. The implemented heuristic method searches for known keywords in the text. Knowledge about didactic functions of fragments is for instance used for modularization methods (see Section 3.2).

### Content Modifications

According to the concept from Section 5.3, changes of the contents of learning resources are not performed in the SCR. The reason is that changing the SCR directly would require to transform it back into the original document format afterwards, what would most probably lead to an information loss. Instead of immediate changes, the re-purposing tools only specify modifications with regard to the RDF entities of the Semantic Content Representation. A modification is a local, relatively small change of the learning resource. Each modification is passed to a so called Modification Transaction Engine(MTE) for processing. The MTE finds out which document format plug-in is responsible for the concerned elements and forwards the modification to that plug-in. The document format plug-in is finally responsible for the execution of the modification.

Modifications have been chosen to be modeled as Java objects for the implementation. There is a top level interface *IModification*, which has to be implemented by all modification type classes. Each modification type is represented by its own class. In total, 23 different modifications have been implemented. Implemented modification types are, for example, replacement of text fragments, insertion of images, movement of elements to another position, changes of text or background colors, and splitting one module into multiple modules.

The Modification Transaction Engine performs two important actions before a modification is passed to the responsible document format plug-in. A re-purposing tool always specifies which content elements are affected by a modification. But the tool knows only the RDF identifier from the SCR. As mentioned above, the MTE holds a hash table that maps these SCR identifiers to the corresponding Java objects from the OOCR. The MTE reads the SCR identifiers from a submitted modification, looks up the corresponding Java objects and writes references to these objects to the modification object. Thus, subsequent processes can immediately access these objects that are the subject of the modification. One of the affected OOCR objects is regarded as the main object. The MTE determines to which document format plug-in this main object belongs; this document format plug-in is responsible for executing the modification. Finally, the modification object is passed as an argument to the execution method of the document format plug-in.

### Document Format Plug-ins

The re-purposing framework provides two mechanisms of abstraction: the Semantic Content Representation and content modifications. Both mechanisms relieve applications from knowing details about a particular document format. Mappings between an abstracted view and a particular document format are outsourced to document format plug-ins. For each document format, which the re-purposing framework should support, a separate document format plug-in is required. The format plug-in provides all functionality that is required to handle documents of the particular format according to the framework concept. A document format plug-in is integrated into the re-purposing tool suite as an Eclipse plug-in. An extension point for format plug-ins is provided by the re-purposing framework. As soon as a new document format plug-in is made available as Eclipse plug-in, the re-purposing tool suite recognizes this plug-in and is enabled to process the new document format.

A new document format plug-in has to implement basically four features. These are

- The document format plug-in is realized as a Eclipse plug-in. It has to extend the given extension point *com.sap.research.cs.ModulEditor.formatPlugIn*. Furthermore, there has to be a class present which implements the interface *IFormatPlugIn*.
- The class mentioned above provides a method *generateOOCR()* for generating the OOCR objects for a given document. The overall document is represented in the OOCR by an object implementing the *IDocument* interface. This object typically contains several children, which implement further OOCR interfaces.
- For the generation of the SCR, two methods are required for each OOCR object. The first method provides a list of child objects for the tree traversal. The tree traversal algorithm automatically inserts a node into the SCR for each OOCR element. The second method, *addPropertiesToSCR()*, can be used to add a description of the element to the SCR. Generally, at least a type from the Content Ontology is assigned to the node to ensure proper processing by re-purposing tools.
- Each document format plug-in has to provide a method *executeModification()* that performs modifications of contents of the supported document format.

In the current implementation, the formats SCORM, HTML, CSS, and an internal XML-based document format of SAP AG are supported by the Re-Purposing Tool Suite by document format plug-ins. Images and other media objects are not supported by document format plug-ins, but only as references to binary files. Re-purposing tools have to process images directly without dedicated support of the framework.

### 7.2.3 Modularity-Aware SCORM Editor

The Re-purposing Tool Suite has a built in SCORM editor (see Figure 7.6). The editor is the central view of the Re-purposing Tool Suite. All further actions can be triggered from the SCORM editor. The editor has three view areas: a module hierarchy view, a module editor view and a property view. The module hierarchy view shows a tree of all modules that a loaded learning resource is composed of. From the module hierarchy view, re-purposing actions can be invoked. The module editor view enables to view

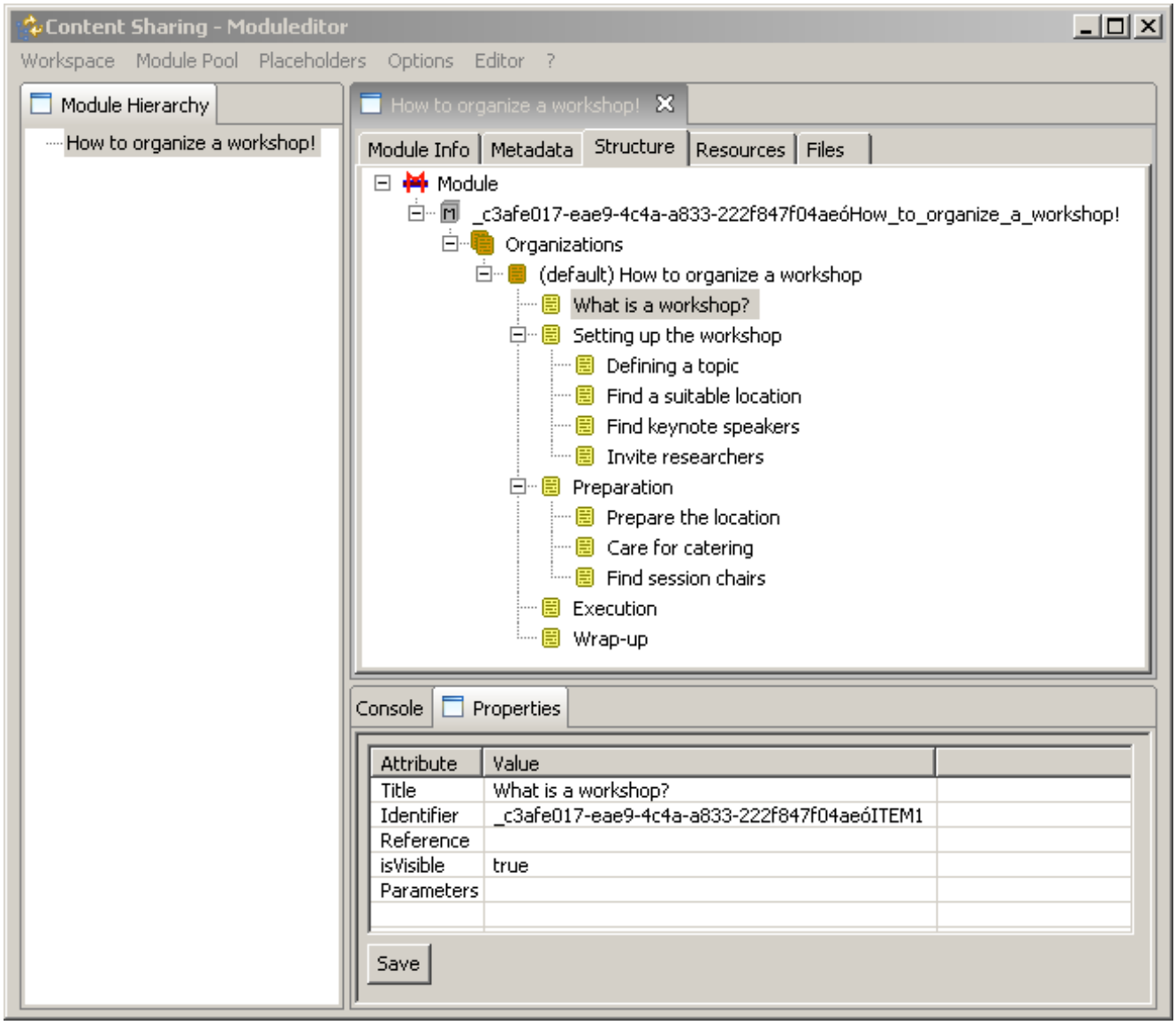


Figure 7.6: Re-purposing Tool Suite with integrated SCORM editor.

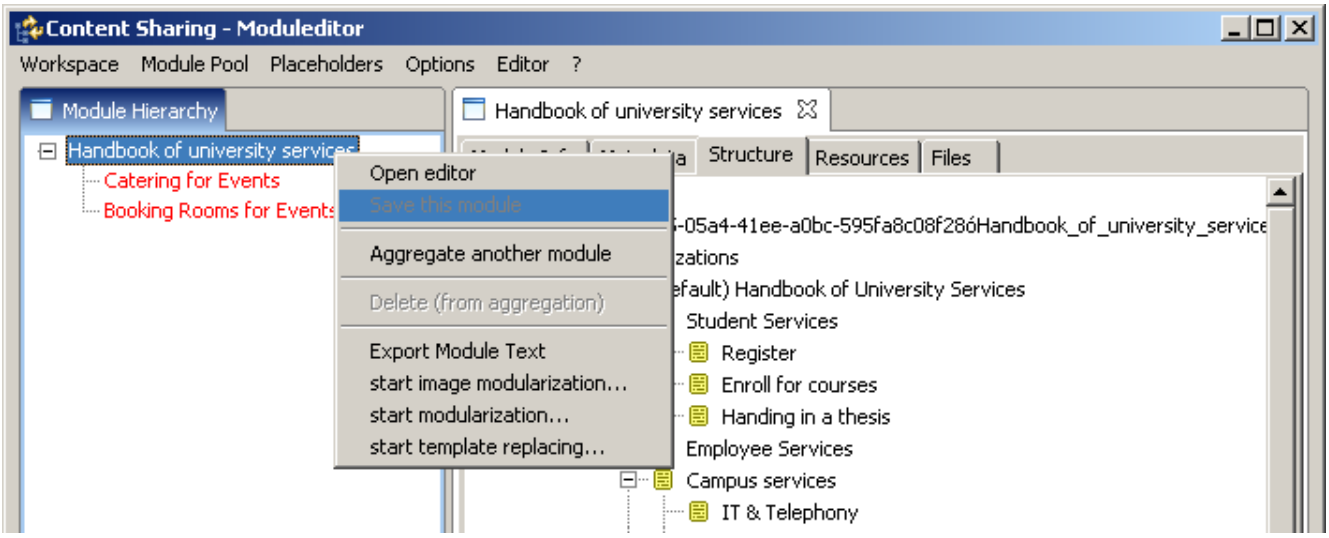


Figure 7.7: Invocation of a re-purposing tool via the module context menu.

and edit the data of one module. Multiple editor instances can be opened at the same time. Each editor view has five tabs between which the user can switch. The tabs show general module information, the metadata record of a module, the organizational structure of the SCORM manifest, resource definitions from the manifest, and a flat list of files that are contained in the module folder. Properties of structural elements, resources and files can be displayed and changed in the property view.

The editor allows to view and edit the structure of a module's SCORM manifest. But a SCORM module contains further content documents. In order to view and edit these content documents, the default viewer and editor of the local system can be invoked to open a content document.

The SCORM editor fully supports the module concept and thus is able to display and change aggregations. A high-level view of the module hierarchy of a learning resource is shown in the module tree on the left hand side of the editor window. The editor supports the organizational level of the aggregation concept – besides the standard SCORM features, organizational items may also link into manifests of included modules. These links are visually differentiated from normal items and can be edited in the SCORM editor.

In fact, the modularity-aware SCORM editor is not just a simple editor, but actively supports two of the six modular operations specified in Section 2.2.2: exclusion and reorganization. Exclusion means to remove modules from an aggregation. The editor provides a function that removes included modules from the overall structure and fully cleans the manifest and the metadata record of the aggregating module. Reorganization means to change the order of modules within an aggregation. The position of an included module is specified by the organizational structure of the referencing module. The module-aware SCORM editor enables reorganization of modules by reorganizing the SCORM structure of the referencing module.

### 7.2.4 API for Re-Purposing Tools

An application programming interface (API) was defined for the integration of re-purposing tools into the Re-purposing Tool Suite. The API specifies how a tool is made visible to the framework, how it is invoked by a user from the tool suite and how the tool interacts with the data model of learning resources.

Basically, the integration of re-purposing tools works analogue to the integration of document format plug-ins. A re-purposing tool has to be developed as an Eclipse plug-in and attached to an extension point. The tool plug-in is required to provide and expose a class that implements the *IRepurposingAction* interface. This interface consists of three methods: two methods can be used by the tool suite to obtain a label and a tool tip text from the plug-in. The third method actually invokes the tool and takes the affected module as an argument.

The re-purposing tool is invoked from the module hierarchy view in the main window. Each module has a context menu, which contains the available re-purposing actions (see Figure 7.7). All available re-purposing tools are dynamically inserted into the context menu. A re-purposing action can be focused on any of the loaded modules. Discussions with users have disclosed that it is not sufficient to perform re-purposing processes on the whole aggregation. Especially if a learning resources has been composed of different modules that require certain adaptations to fit together, the adaptations should be restricted to particular modules. For instance, if a module written in Spanish is aggregated into an English learning resource, the language adaptation should operate only on the Spanish module.

Once invoked, the re-purposing tool can freely operate on the target module. However, there is an interface that facilitates the analysis and modification of the contents. These interfaces particularly provide access to the Semantic Content Representation and the Modification Transaction Engine (cf. 7.2.2). The SCR can be accessed via the native interface of the Jena RDF library. The tool developer may chose between a navigation on the RDF graph and RDF queries. For performing changes of the contents, the tool creates modification objects and passes them to the MTE. Modification objects and their treatment by the framework have already been explained in the previous section.

## 7.3 Implemented Re-Purposing Tools

The previous section has introduced the overall Re-purposing Tool Suite. The content representations, plug-in mechanisms for document format plug-ins and re-purposing tools have been explained. The module manager and the SCORM editor have shown the basic editing functionality that is provided by the Re-purposing Tool Suite.

This section now focusses on three particular re-purposing tools that have been built on top of the framework and that have been integrated into the tool suite. The three tools are a modularization tool, an aggregation tool and an adaptation tool. The first two tools are based on concepts from earlier chapters of this thesis. The last tool has been designed and implemented by a colleague, but is described for the sake of completeness and to demonstrate the functional capability of the re-purposing framework.

### 7.3.1 Interactive Modularization Tool

There are very different potential modularization methods, as Section 3.2 has already proven. For the particular scenario of the Content Sharing project, two instances of the generic modularization process have been realized. The two implementations are very different in their approach and functionality, which supports the generic claim of the reference process. The first implementation supports an interactive modularization process, which transforms organizational elements from the SCORM manifest into separate modules. The second modularization method concentrates on media elements, which can be extracted from the learning resource as separate media modules. The main focus of this section is the first method, the second one will be described only briefly.

#### Scenario Requirements and Modularization Concept

Section 3.2 has synthesized a generic modularization process out of existing descriptions of modularization methods. This generic modularization process has been mapped to the present scenario and conditions. As a result, a concept for an interactive modularization tool has been created, which can be used for modularization of SCORM-based learning resources by content producers and content re-users. The modularization tool concept has been created and revised based on several interviews with both, content producers and content users from the Content Sharing project. A modularization tool for this target group should be able to deal with modular learning resources according to the module concept of this thesis. Users want to get as much support as possible from a tool; it is desired that the tool proposes reasonable module boundaries. On the other hand, they want to be able to freely adjust module boundaries. It is a challenge to find a balance between simplicity and freedom of choice. And finally, users from the target group have asked for support in metadata generation and handling. An often mentioned demand is to let the modularization tool propose as much metadata as possible. This also matches with the observation made by Hoermann [Hoe05]. Three out of the six steps of the modularization process have been designed as interactive tasks: preprocessing, content analysis, boundary determination. Technical decomposition and postprocessing shall be performed without user intervention. The planning phase will not be technically implemented; planning has to be done by the user in advance.

An approach to match the users' demand for support and yet a certain freedom of choice is to initially propose module boundaries and afterwards allow the user to alter the suggestion if needed. This shall be achieved by presenting an outline of the course structure. The outline is presented to the user as a tree, combined with meaningful descriptions of the elements. Proposed target modules are supposed to be visually distinguished to enable the user to overview and understand what will be the result of the modularization. Proposals for reasonable module boundaries shall be provided by the modularization tool; boundaries could for instance be proposed based on the depth of elements in the tree. These boundaries should though be freely editable afterwards by the user. Additional support should be provided to the user by giving him more information on the contents of individual structural elements. This information can either be obtained from existing metadata or by performing a content analysis. When the user has confirmed the determined module boundaries, the modularization tool should automatically decompose the learning resource according to the chosen boundaries. The modularization process shall result in an

aggregation of multiple modules. The overall learning resource will be still complete, but will consist of multiple modules that can be reused separately.

The tool concept also includes the utilization of metadata strategies. A metadata strategy is a replaceable method which generates a new metadata record out of a given input. In this case, the input consists of the metadata record of the original learning resource, the contents of the new learning resource, and context information (such information about the user, the system and the modularization process, which has been performed). An interactive metadata strategy could also use a dialog for obtaining additional information from the user or to verify metadata proposals.

### Implementation

Based on the above concept, a modularization tool has been implemented. The modularization tool has been integrated into the Re-purposing Tool Suite, using the mechanisms described in the previous section. The tool uses the Semantic Content Representation from the re-purposing framework, semantic content enrichment methods, and a modification type for modularization. The modularization tool has been designed as a so called wizard (an application, which leads the user through a strict sequence of dialogs). How the five implemented process steps have been realized in this process instance is described in the following paragraphs. The five consecutive dialog screens are illustrated in Figure 7.8.

- Preprocessing is the first technically supported phase of the implemented modularization process. Preprocessing means to transform the learning resource into a state, which is better suited for modularization. In the present implementation, the assignment of files to resources of the SCORM manifest is completed. The motivation for this step is that the SCORM specification is too lax concerning the resource definition. The usage of file elements is optional, which leads to SCORM packages in which no explicit connections between files and resources or items are available. But a modularization can only be successful, if all required files are finally located within the right module. To relieve the actual decomposition method from these concerns, file assignments are determined and made persistent in an early stage.

- In a second process step a content analysis takes place. Goal of the content analysis is to provide the user with more information about the contents of structural elements, such as SCOs and assets. Again, implicit information is made explicit for simplification. The implemented content analysis method determines the didactic type of each element. For instance, elements can be automatically tagged as introduction, definition, or example. The didactic type is written to the Semantic Content Representation as an annotation and will be available in subsequent process steps. Another content analysis method is the estimation of the granularity level of elements (see Section 6.5).

- The third step of the wizard is the core dialog page of the modularization process. The user is given an abstract outline of the learning resource. Module boundaries are determined in this view. The learning resource outline is enriched with the semantic annotations that have been generated in the previous process step. There is a slider for the user to control the granularity of boundary proposals. The user may select the structural depth at which new modules are proposed. Each target module is assigned a unique color to help the user to distinguish the modules. When the

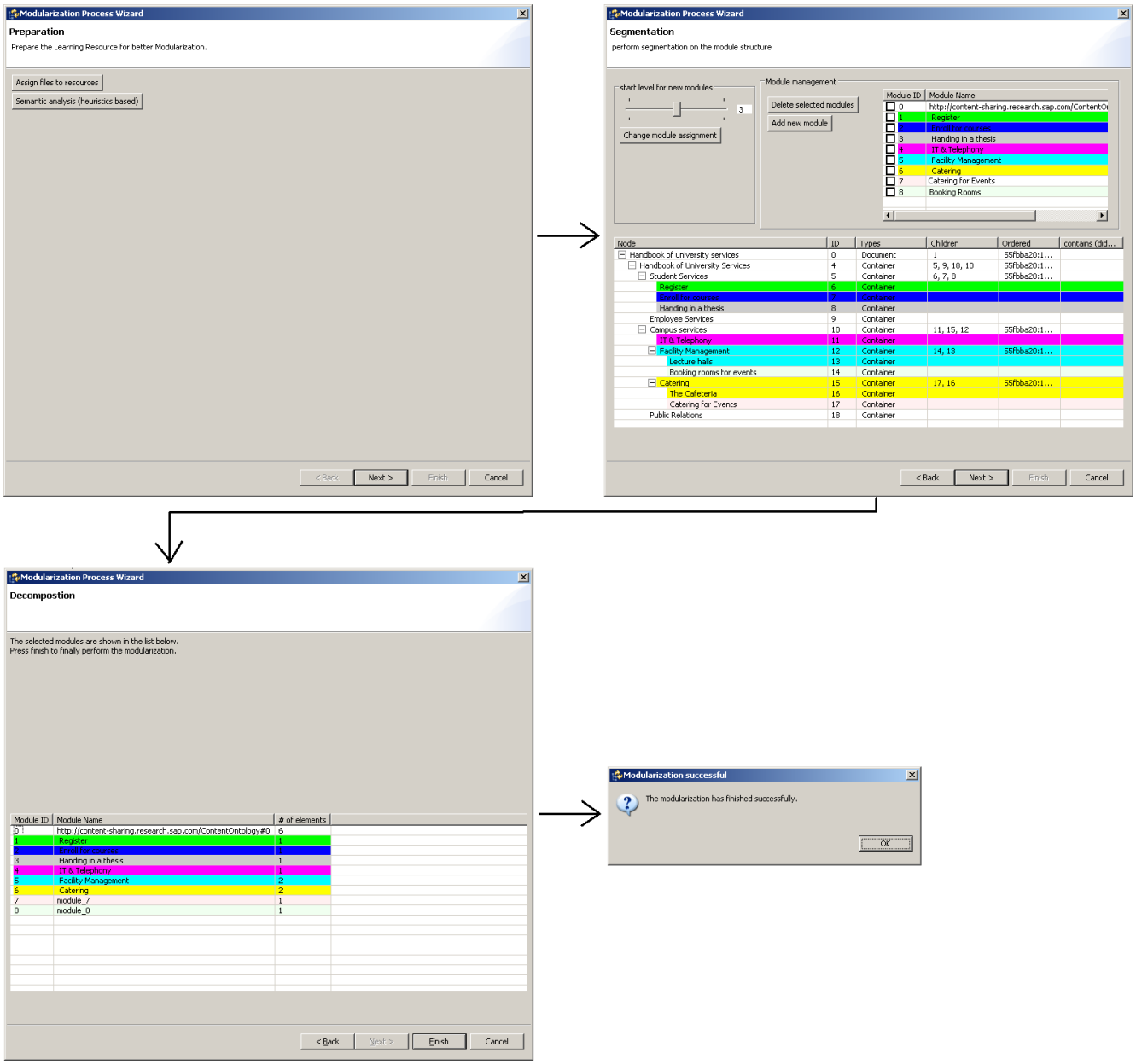


Figure 7.8: Interactive modularization wizard.

user moves the slider, new module boundaries are proposed. He may afterwards add or delete target modules and reassign individual structural elements to different target modules. The title of the first structural element in a target module - if such a title exists - is proposed as the title for the module; in practice, this rule of thumb has proved to satisfy the users. All in all, the implemented boundary determination view provides a mix of user guidance and freedom of choice. The user is not restricted in his choice of module boundaries, but may benefit from the interactive support.

- After the user has confirmed the chosen module boundaries, the physical decomposition takes place. As all dependencies between structural elements and files are already known from the preprocessing step, the structural elements are moved to newly created target modules. If a file is required by more than one structural element, it is copied instead of moved. The decomposition is solved as a modification command as described in Section 7.2.2. The modularization modification object is passed to the SCORM document format plug-in of the re-purposing framework for execution. At this point, the functionality of the framework can be fully exploited: how the decomposition is technically implemented is the responsibility of another component and can even be changed without changing the modularization tool as such.

- As last step of the decomposition process, a metadata strategy is applied to create metadata records for the new modules. Currently, a simple non-interactive metadata strategy is used. The method copies all those fields from the original metadata record to the new module, which are supposed to remain valid. A new title is estimated as described above. The application also captures some lifecycle information and writes it to the metadata record. The lifecycle information can be used to track the evolution of different versions of a learning resource in a central metadata repository [LHRS07].

After the modularization process has been completed, the learning resource has been transformed into an aggregation of multiple modules. The resulting modules can also be exported separately and aggregated to new courses with different contents.

### Modularization of Media Objects

Discussions with potential users have revealed that users do not always want to modularize whole text passages or pages. Often, embedded media elements, such as images, videos, or interactive Flash animations have a high value and are desired to be reused independently of the surrounding learning resource. Therefore, a media extraction tool has been implemented as an alternative modularization method.

In this tool, the generic modularization process is instantiated differently than before. The basic idea is that all contained media elements are presented in a flat list. The user may browse this list, view previews of the media files and select those media objects, which he wants to have as separate modules. Only four process steps are technically supported, as the following description shows.

- The preprocessing step is analog to the first modularization tool: All files are assigned to one or multiple resources of the SCORM manifest. Especially the media files have to be clearly assigned, because they have to be moved to another module afterwards.

Externalize MediaElements Wizard

Detect and Select Media Elements

Existing Elements

| Filename | Type | Relative Path | Filesize | Path |
|---|---|---|---|---|
| Qualitat2.jpg | jpg | 0bbbe951-fe91-403e-aba8-30683c4082a6\3416\bilder | 8 KB | C:\Doc |
| start2.gif | gif | 0bbbe951-fe91-403e-aba8-30683c4082a6\3391\bilder | 92 B | C:\Doc |
| Simultan1.gif | gif | 0bbbe951-fe91-403e-aba8-30683c4082a6\3416\bilder | 364 B | C:\Doc |
| netz.gif | gif | 0bbbe951-fe91-403e-aba8-30683c4082a6\3409\bilder | 2 KB | C:\Doc |
| monitor.gif | gif | 0bbbe951-fe91-403e-aba8-30683c4082a6\3412\bilder | 14 KB | C:\Doc |
| Rechtsbundig2.jpg | jpg | 0bbbe951-fe91-403e-aba8-30683c4082a6\3391\bilder | 55 KB | C:\Doc |
| Warmkalt2.jpg | jpg | 0bbbe951-fe91-403e-aba8-30683c4082a6\3416\bilder | 5 KB | C:\Doc |
| cmyk.jpg | jpg | 0bbbe951-fe91-403e-aba8-30683c4082a6\3424\bilder | 12 KB | C:\Doc |
| Blocksatz.jpg | jpg | 0bbbe951-fe91-403e-aba8-30683c4082a6\3391\bilder | 46 KB | C:\Doc |

View Element

Select Element to Externalize

Selected Elements

| Filename | Type | Relative Path | Filesize |
|---|---|---|---|
| Ausschr.jpg | jpg | 0bbbe951-fe91-403e-aba... | 23 KB |
| Schriftart.gif | gif | 0bbbe951-fe91-403e-aba... | 6 KB |
| Fliesstext.jpg | jpg | 0bbbe951-fe91-403e-aba... | 21 KB |
| Komplementar3.gif | gif | 0bbbe951-fe91-403e-aba... | 492 B |

Deselect Element to Externalize

< Back | Next > | Finish | Cancel

Figure 7.9: Modularization of media objects.

- As content analysis step, all references from content pages – typically HTML documents – to media files, such as images, are identified. This list of references serves two purposes lateron. First, it is used to display the list of media objects to the user. And second, the references in content documents have to be known in order to change these references to the new file locations after the modularization.
- In the boundary determination step a list of all identified media objects is generated and provided to the user, as shown in Figure 7.9. Some properties are listed, for instance the file type, size, and path. The user may also open a preview of the media file in order to get a better understanding of the contents. The user selects all files from the list, which he wants to transform into separate modules.
- In the decomposition phase, the previously selected media objects are turned into separate modules. In the present implementation, each selected media object is transformed into an individual module. The new modules are aggregated into the original module by reference. The references from content pages (e. g. HTML) to the media files are updated with the new relative path.

Again, a planning phase is not supported by the tool. A post processing step has also not been technically implemented. The change of references has already been performed in the decomposition step, so it is unlikely that the decomposition produces flawed modules. Metadata generation could take place – at least some technical information about the media objects could be determined and written to the metadata record. For the current implementation, using the file name as module title is considered to be expressive enough.

#### Conclusions

Both modularization approaches, the modularization wizard and the media extraction tool, are instances of the generic modularization process from Section 3.2. A comparison of both approaches demonstrates the wide variety of possible implementations of the generic modularization process. The two instances are based on differing requirements and expectations and thus also differ in their implementations.

From a functional point of view, the implementation shows that SCORM based learning resources can be modularized as designed by the modularization concept. It is possible to decompose learning resources and to exchange them via interchange packages.

### 7.3.2 Aggregation

Another of the modular operations introduced in Chapter 2 is aggregation of modules. Aggregation means the combination of multiple modules in order to create a larger module. Chapter 4 has examined aggregation from a theoretical point of view. As a result, existing authoring by aggregation methods have been improved towards a new authoring by aggregation process. This process has been realized in the Re-purposing Tool Suite. The functionality is distributed over several components of the tool suite.

## Concept for an Aggregation Tool

Chapter 4 has examined existing approaches for authoring by aggregation. As a shortcoming of existing approaches the missing dedicated support for the didactic design of a learning resource has been identified. An improved authoring by aggregation process integrates didactic design as a separate authoring phase. There are two variants of authoring by aggregation process: a sequential version and a parallel version. The strict, sequential process variant specifies that five authoring phases have to be performed one after the other – strictly realizing the conceptual considerations (see Figure 4.4). A relaxed version of this process (figure 4.5) abandons the strictly sequential order because of practical considerations. It is likely to happen that a user realizes in one phase that he has to change the work of a previous phase; therefore the theoretical assumption of completing one phase before starting the new one has been substituted by the acceptance of overlapping authoring phases. Thus, the user has at any point of time the choice to perform actions of any of the four authoring phases.

The relaxed authoring by aggregation process has been implemented in the Re-purposing Tool Suite. The tool suite user interface enables different actions for the four authoring phases *didactic design*, *retrieval & replacement*, *adaptation* and *content authoring*. The fifth phase, *publishing*, is also offered at any time, but conceptional ends the authoring of one learning resource. The authoring phases are designed as follows.

- **Didactic Design.** The didactic design of a learning resource is supported by the introduction of placeholders. The user may create an empty structure for the learning resource and replace the placeholders later with existing modules or new contents. Placeholders are created and edited like other elements of a SCORM manifest in the SCORM editor view. Details about placeholder items can be found below.

- **Retrieval & Replacement.** Placeholders can be either replaced by existing modules, or filled with new contents. The replacement of a placeholder by a module can be performed by the user via a replacement wizard. This wizard is integrated into the tool suite as a re-purposing tool using the known mechanisms.

- **Adaptation.** Adaptation of modules is enabled by a separate adaptation tool. The adaptation tool can be invoked from the context menu of the module hierarchy view. More details can be found below in Section 7.3.3.

- **Content Authoring.** Creating and editing a learning resource affects not only the structure but also the content documents. The Re-purposing Tool Suite integrates existing editors for content documents into the SCORM editor view. From the context menu of an SCORM item or resource the default editor and viewer of the host system for the particular document type can be invoked (e. g. the user's default HTML editor is invoked for editing an HTML document).

- **Publishing.** The publishing phase of a new learning resource has already been described in Section 7.2. The involved modules are transferred to the local module pool. From there, they can be exported as an interchange package and disseminated.

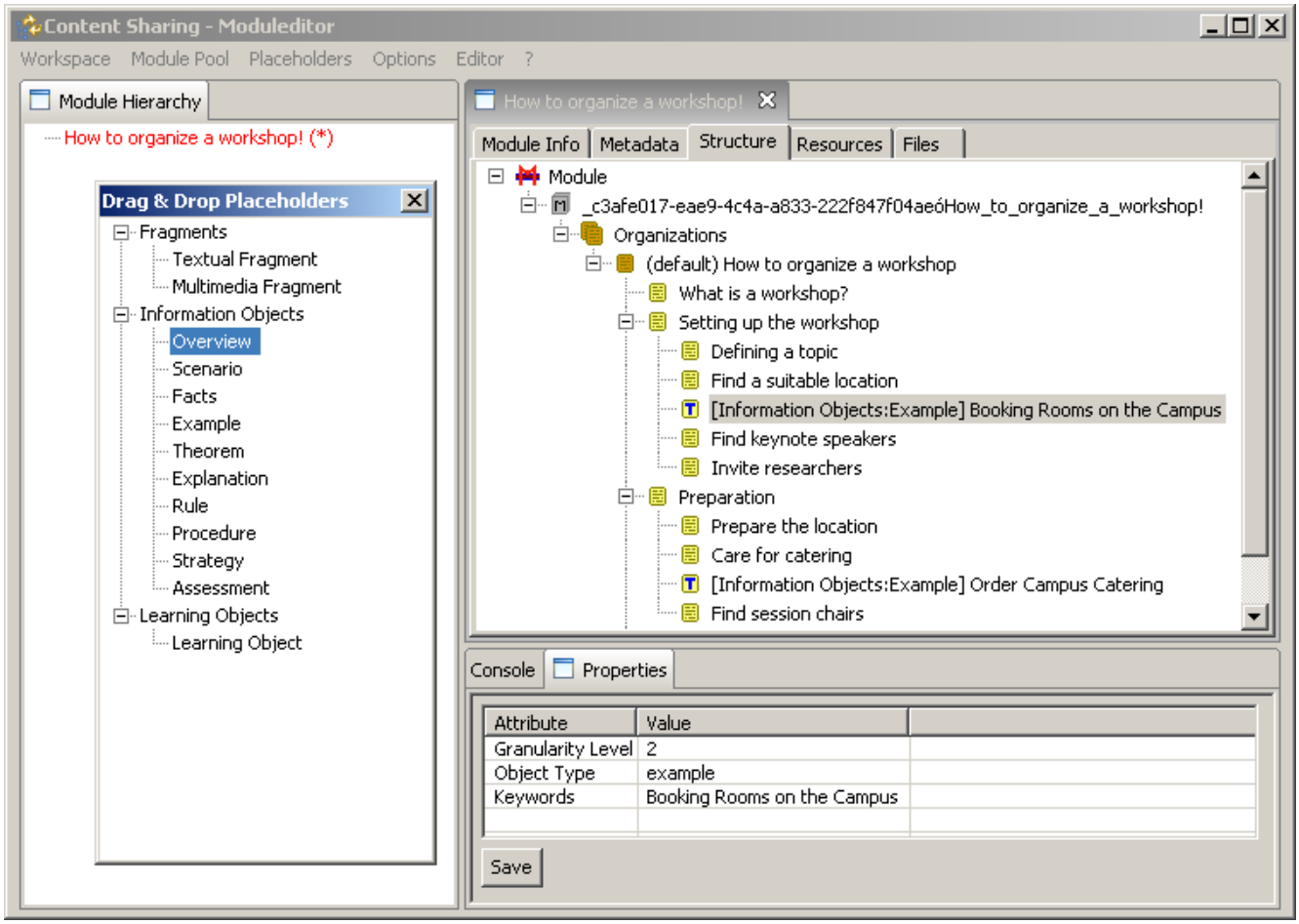


Figure 7.10: Drag & drop menu for placeholder types.

## Placeholder Generation and Editing

In the didactic design phase the user may create placeholder elements in the organizational structure of a learning resource. The placeholders are implemented as additional tags in the SCORM manifest. These *placeholder* tags do not comply to the SCORM standard and are therefore only intended for internal use – placeholders cannot be exported, but have to be completely replaced before. A placeholder element can be placed at any position where an *item* element is allowed. It has three attributes: granularity level, object type and keywords. The granularity level attribute complies with the aggregation level field of LOM; it indicates the size of an element. Object types are mostly used for information objects (aggregation level: 2). An object type specifies the didactic kind of information object the user intends to use. This attribute matches the *learning resource type* field of LOM, but uses in the present implementation another vocabulary. The keywords attribute, finally, enables users to specify briefly in words the planned contents of the new element. Keywords serve as a reminder for content production, or as query terms for the retrieval of existing modules.

Placeholder elements are created and edited in the SCORM editor view like other elements of the learning resource structure. New elements are inserted via a context menu; attributes of a placeholder can be edited in the properties view. Additionally, an object types palette is offered, which enables users

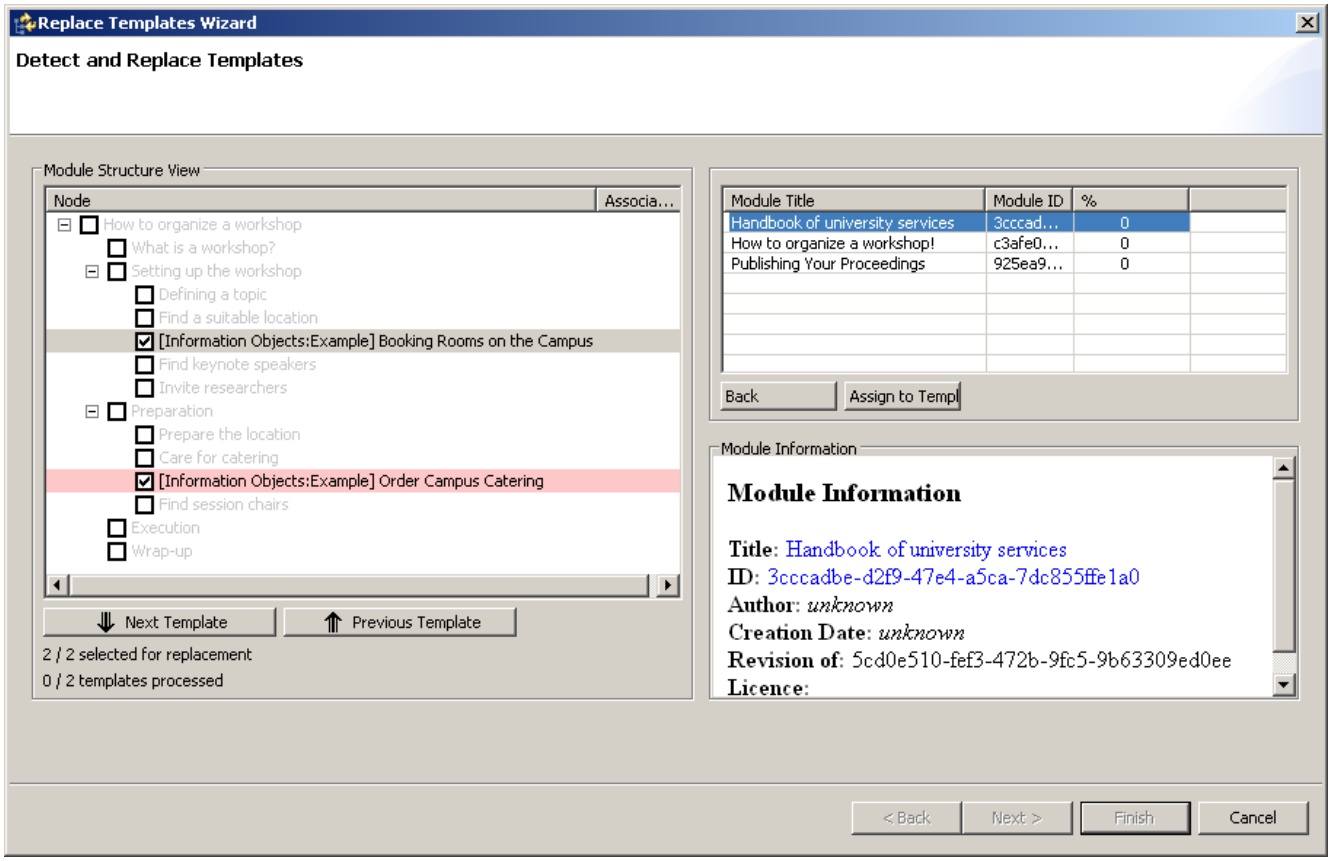


Figure 7.11: Retrieval and replacement of placeholder elements.

to drag & drop an object type from the palette into the SCORM editor view (see Figure 7.10). The drag & drop approach shows the available object types to the user and allows to create a new learning resource structure intuitively.

The attributes of a placeholder element can be changed after the placeholder has been created. The granularity level, object type and the keywords are displayed in the structure tree of the editor view.

## Replacement Wizard

In the retrieval & replacement phase of the authoring by aggregation process the user may substitute placeholders by existing modules from a repository. This replacement is realized again as a wizard, which is plugged into the Re-purposing Tool Suite as a re-purposing tool.

According to the concept from Chapter 4 the user can chose for each placeholder to replace a placeholder by a suitable existing module or to create new contents. Replacing a placeholder by a module requires that the user queries for suitable modules; only if a fitting module has been found it can substitute the placeholder. The retrieval of modules is performed sequential one after the other. The user needs knowledge about the placeholder for which he searches a substitute. The three attributes of a placeholder element might not suffice for orientation; a user can perform the query task best if he knows also the environment of the element – what comes before and after that element. Therefore, the replacement

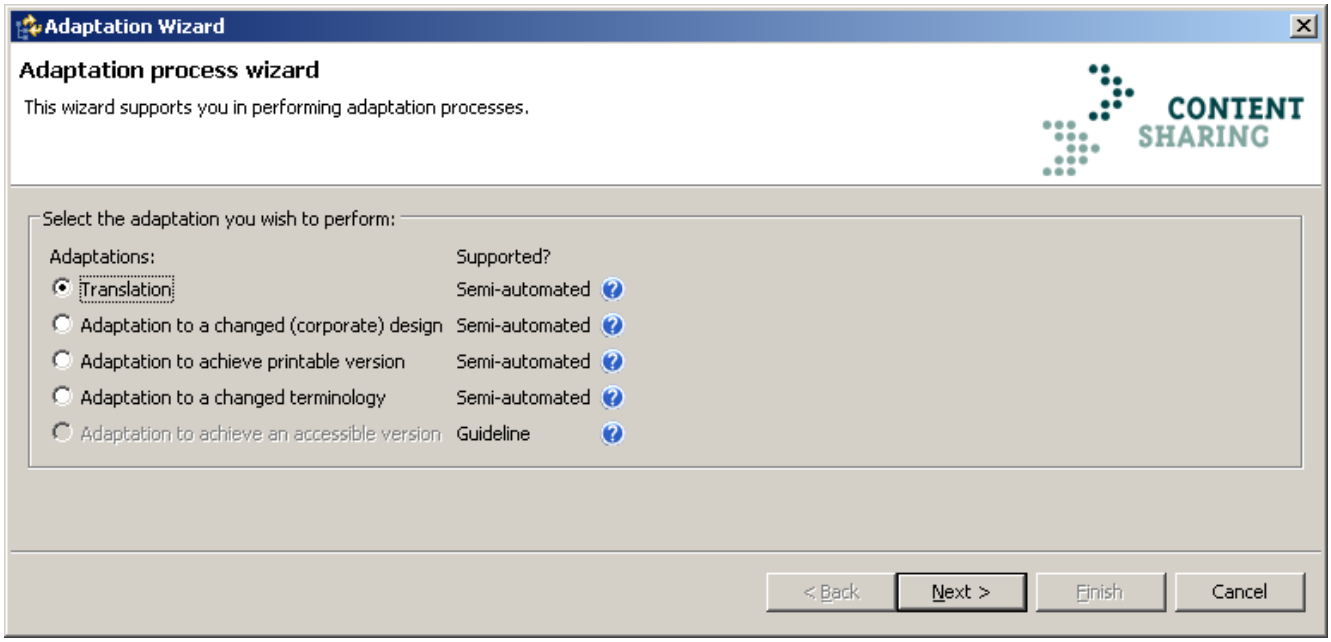


Figure 7.12: Adaptation of learning resources.

wizard begins with displaying the structure tree of a learning resource again. Placeholder elements are highlighted in color (see Figure 7.11). When the user selects one of these placeholder elements in the tree, the attributes of this placeholder are shown and can still be edited. The repository (respectively the local module pool in the present implementation) is queried for fitting modules using the placeholder attributes as input. The LOM fields aggregation level and learning resource type have to match the corresponding placeholder attribute. The entered keywords are compared to textual fields of the metadata records, such as title and description. A ranked list of found modules is displayed. For each module an abstract of the metadata record can be viewed. If the user finds a suitable module among the results, he can assign it as substitute to the selected placeholder. This procedure is repeated for all placeholders of the learning resource structure. If a user decides to not replace a placeholder element by an existing module, he has to actively deselect the placeholder before he can proceed to the next wizard page. This obstacle has been built in after a first usability check in order to prevent that the user misses some placeholders and has to start the replacement tool several times again.

From a technical perspective, the replacement of a placeholder by a module performs an aggregation corresponding to the definition of modular operations. The substitute module is copied from the module pool into the workspace. The placeholder element is replaced by a regular item tag, which references the manifest identifier of the substitute module. The chosen module is fully aggregated by adding an aggregation relation to the metadata record of the aggregating module and connecting both manifests by an *xinclude* tag.

### 7.3.3 Adaptation

The third implemented re-purposing tool is an adaptation tool. The adaptation tool is based on an analysis of common adaptation tasks and how they are performed by users. Zimmermann et al. have identified 15 relevant adaptation processes [ZBRS06]. Five out of these 15 have been chosen for implementation in the Re-purposing Tool Suite [ZRS07]. The implemented adaptations are performed by a single re-purposing tool, which can be invoked for any of the loaded modules from the context menu of the module hierarchy view. Currently, following five adaptation types are supported: adaptation to a new (corporate) design, making a learning resource compliant with Web accessibility standards, optimization of printability, translation into other languages, and the adaptation of terminology.

The chosen adaptation processes modify content documents and do not change the overall SCORM structure of a learning resource (except the substitution of images). As the contents of documents are the focus of interest, the content representation model is more extensively used by the adaptation tool than by the other described tools. The adaptation tool is designed to handle different document formats. Furthermore, the adaptation processes operate always on a whole learning resource, although it typically consists of several documents. For instance, design changes, such as changing background colors and images or font styles, are performed per module and not per document. The advantage is that the appearance is coherently changed for all documents of a module.

Adaptation processes have been organized by Zimmermann into three levels of task granularity: a whole *adaptation process* is divided into several *process fragments*[50], which again make use of multiple *adaptation functions* [ZBRS06]. This internal structure of adaptation processes also reveals how adaptations are integrated into the tool suite concept. The overall adaptation tool, which offers the five implemented adaptation processes, can be invoked for a module from the module hierarchy view. An adaptation process is realized as a wizard, which guides the user step by step through the whole process (see Figure 7.12). Process fragments are mapped to individual wizard pages. The smallest parts of the process model, adaptation functions, are small tasks that either obtain information about the learning resource, request a decision from the user, or modify the contents. The first type, obtaining information, is realized by querying the Semantic Content Representation. The execution of semantic enrichment components can also be considered as such an adaptation function. Adaptation functions that change a learning resource are mapped to the modification concept, which has been introduced in Section 5.3. Thus, a process fragment is an interactive program part, which sequences content queries, decisions, and content modifications. According to the content representation concept, the adaptation tool is format independent insofar that it can deal with contents in all document formats that the re-purposing framework supports. The development process for the adaptation tool is described in detail in [Zim08].

## 7.4 Evaluation

The prototype described in this chapter implements the concepts for the base modular operations from the previous chapters of this thesis – modularization, aggregation and adaptation of learning resources. These concepts are proposed solutions for achieving modularity of learning resources and especially

[50] Process fragments are also called process steps in newer publications.

multi-granularity reusability. The question remains if the requirements for modularity and multi-granularity reuse are fulfilled by the implemented prototype. The first part of this section reflect the implementation process of the Re-purposing Tool Suite and feedback from users within the Content Sharing project. The second part compares the requirements from the Chapters 2 and 3 with the implementation in order to find out if the requirements could be achieved.

### 7.4.1 Lessons Learned from Development and Feedback

Over the duration of the Content Sharing project, several milestone versions of the prototype have been rolled out to selected users. The users have tested the prototype in their typical work environment. Besides a free test of functionality, the users were also given certain tasks, which they had to perform. Because of the low number of users, a statistical evaluation was not feasible. A detailed user study for the adaptation tool is currently conducted by a colleague. However, the previous feedback provided a qualitative assessment of the prototype in the consecutive development stages.

The result of the test were, that the latest version of the implemented prototype actually offered the expected functionality. Users were able to import, modularize and aggregate SCORM-based learning resources. They were also able to perform the supported adaptations on learning resources, which contained HTML documents as content documents.

The reactions from the users indicated that the usability of the prototype has yet a potential for improvements. Especially invalid user actions should be detected or prevented earlier.

However, the most important result of the usability test has been the identification of a new user group: the non-authors. Non-authors are users who are not educated for content authoring and therefore do not have a technological background, such as knowledge about SCORM, HTML or image formats. When the user group of reuse systems expands to also cover non-authors, new requirements for reuse tools arise. Modularization and aggregation have to become more intuitive. Technical details, such as the SCORM nomenclature, have to disappear or to be replaced by colloquial language. New metaphors need to be found for enabling non-users to naturally handle these tools.

This finding is notable, because the scenario of multi-granularity reuse apparently attracts potential users, who before did not engage in content authoring. Those users previously simple reused existing learning resources unchanged. When re-purposing applications become available, they address a wider user community than only professional authors. Non-professional reusers desire to benefit from these new tools, as well. Thus, re-purposing applications have to meet the technical skills of this broadened user community.

### 7.4.2 Evaluation of Reusability Requirements

For an evaluation of the concepts, which have been realized in the prototype, the requirements for enabling modularity of learning resource and multi-granularity reuse are compared to the functionality of the prototype. The proposed concepts are valid only if they – respectively their implementation – enable modularity and multi-granularity reuse. This subsection focusses on functional aspects of the prototype. Usability aspects have already been addressed in the previous subsection.

### Modularity

The first evaluation is the realization of modularity of learning resources. The definition of modularity in Section 2.2.2 consists of two necessary requirements towards a learning resource specification: first the support of encapsulation, exposition and separate reuse of parts of a learning resource; and second the availability of six modular operations on modular learning resources.

The first requirement, encapsulation, exposition and separate reuse of parts of a learning resource concerns the property of a learning resource specification to statically allow a distribution of contents over several modules, and particularly to treat these modules also as self-contained, reusable units. This property is fulfilled by the implemented modular learning resource format. Contents of a learning resource can be indeed distributed over multiple modules, because a learning resource can be represented as an aggregation (by reference) of modules. Each of the modules, which are part of a whole learning resource, is again valid learning resource: it can be exported as an individual, self-contained SCORM package; and it can be aggregated into different learning resources.

While the first requirement resembles a static property, the second one concerns dynamic behavior. A learning resource specification is considered modular, if it supports six defined modular operations. Actually, as a static format specification cannot perform dynamic behavior, support of reusable operators is more a system property than a specification property. Nonetheless, the specification has to ensure that a system can successfully perform modular operations. The six modular operations are evaluated one after the other to check if and how they are realized by the Re-purposing Tool Suite as a reference system.

- **Modularization.** Modularization is supported by two dedicated modularization tools: the main modularization tool, and an additional extraction tool for media files. Both tools can be used to split a learning resource into multiple modules.

- **Aggregation.** The functionality of the aggregation operation is distributed over multiple components of the tool suite. According to the aggregation concept, the planning of an aggregated learning resource is done in the SCORM editor. The actual retrieval and composition is performed using the integrated retrieval & replacement tool.

- **Exclusion (removal of modules).** Removing modules from a learning resource is simply performed by deleting the reference to that module in the SCORM editor. By deleting this reference, all other aggregation information is adjusted.

- **Replacement (substitution).** The replacement of one module by another one has to be executed manually in two steps. First, the old module has to be deleted; and second the new one is included. It would be desirable for further implementations to provide a tool, which facilitates, for instance, to replace a module by a variant or a newer revision of that module.

- **Reorganization (changing the order of modules).** The order in which modules are integrated into the overall learning resource can be changed in the SCORM editor.

- **Adaptation (transformation of a module).** Adaptation is supported by the adaptation wizard. Currently, five of fifteen possible adaptation types are technically supported. The remaining adaptation types are supported by pattern-based guidelines.

It can be concluded that all six modular operations are supported by the Re-purposing Tool Suite. Some of the operations (e. g. modularization, aggregation, and adaptation) are realize more sophisticated, whereas others (e. g. replacement) require more manual work. Nonetheless, all six modular operations can be performed using the Re-purposing Tool Suite.

### Multi-Granularity Reuse

Second, the prototype has to be checked for its compliance with the requirements for multi-granularity reusability. Multi-granularity reusability has been defined as the general ability of reusing learning resources and their reusable fragments at all levels of granularity.

Multi-granularity reuse can be assessed regarding this general definition. And there are also three properties, which constitute more detailed requirements towards reuse supporting systems. First, the general definition is considered. A system, which claims to support multi-granularity reuse, needs to support reuse of learning resources and learning resource fragments at multiple levels of granularity; and furthermore, different types of reuse – namely simple reuse, aggregation and re-authoring – have to be supported. The Re-purposing Tool Suite supports modularization of an existing learning resource – each element within the organizational SCORM structure, and each media element can be transformed into a separate module. Thus, equating *reusable fragments* of the definition multi-granularity reuse with structural SCORM elements and media objects, the condition of reusability at all levels of granularity as separate is met. These modules can be simply reused; they can be aggregated; and they can be re-authored, e. g. by performing adaptations.

Three requirements specify in more detail what makes systems for multi-granularity reuse usable in practice: availability, retrievability and interoperability of modules.

- **Availability.** The Re-purposing Tool Suite supports aggregation by reference: A learning resource can be constructed as a tree structure of modules, where the inclusion of contents of another module is embodied by a reference to that module. Learning resources, which are aggregated in this manner, can be exported and transferred to a repository that is able to handle the constituting modules at the same time as parts of the overall module and as separate modules. The central repository of the Content Sharing marketplace offers this functionality. Thus, the contents of a learning resources can be made available at multiple levels of granularity at the same time.

- **Retrievability.** By making modules available in the way just described, retrievability is already half achieved. Retrievability, though, requires two more properties, that go beyond mere availability of modules. The first requirement is the existence of adequate metadata for each module. Without suitable metadata, the modules can hardly be found. And second, retrieval methods have to make use of metadata about granularity in order to manage the growing amount of modules that come into existence by modularization.

Metadata generation is implemented in the present prototype only to a little extent. However, a new method for generating subject metadata has been presented in Chapter 6. This method could be integrated either into the tool suite or into learning object repositories.

- **Interoperability.** Interoperability regarding multi-granularity reuse particularly means that multiple modules can be jointly used, e. g. by aggregation into a larger learning resource. It is evident that not all modules can be freely combined, because of topical and didactic differences. However, from a technical point of view, modules have to fit together under the condition that the contents are also compatible. Technical interoperability is ensured by the implemented prototype by two means: first by the module handling, which supports aggregation of modules; and second, the adaptation tool supports the transformation of modules towards a unified appearance (e. g. adaptation of design, language, or terminology).

All in all, the requirements of multi-granularity reusability are met by the implementations presented in this thesis. Most of the requirements are covered by the Re-purposing Tool Suite. Additionally, retrievability is improved by the metadata generation methods from Chapter 6.

## 7.5 Summary

Multi-granularity reuse and modularization – as these terms have been defined in Chapter 2 are abstract requirements. The chapters on modularization, aggregation and adaptation have sketched concepts how the abstract requirements can be met in practice. In this chapter, finally, a prototypical implementation of the developed concepts has been described, which realized multi-granularity reuse of modular learning resources.

The prototype application is a tool suite for re-purposing of learning resources. It consists of multiple components. The core functionality provides module handling, a simple editor for the SCORM structure of a learning resource, and a content representation and modification framework, which facilitates the development of re-purposing tools. Further functionality has been integrated into the tool suite as so called re-purposing tools. The implemented tools are a modularization tool, an aggregation support tool and an adaptation tool. The modularization tool splits a learning resource into multiple modules. The aggregation tool provides support for finding and integrating modules into another learning resource. And the adaptation module offers automated support and guidelines for adapting learning resources to new usage purposes.

The prototype is intended to proof the concepts from the Chapters 3, 4, and 5. For evaluating the suitability of the concepts, the implementation of the concepts in form of the prototype has been compared to the requirements for multi-granularity reuse and modularity. The evaluation has revealed that the Re-purposing Tool Suite fulfills all requirements. Hence the proposed concepts are suitable for realizing multi-granularity reuse and modularity of learning resources.

# 8 Conclusions and Outlook

## 8.1 Conclusions

This final chapter provides a summary and conclusions for the thesis. This thesis has dealt with the modular reuse of learning resources within a heterogeneous scenario, under the assumption that learning resources are exchanged among participants who use different authoring systems. In such cases, Web based learning resources are today exported into the SCORM format which is understood by almost any e-learning system (including authoring systems, learning management systems and learning object repositories). One disadvantage is that learning resources are no longer modular after this transformation. A definition even stronger than modularity is multi-granularity reusability, which has been specified in this thesis. Multi-granularity reusability in particular allows any relevant fragment of a learning resource to be reused as a module. SCORM does not fully satisfy the requirements of modularity and multi-granularity reusability in this way. However, the SCORM specification has been an important milestone in the development of e-learning systems because it ensures compatibility. The specification of a completely new learning resource format would break the compatibility already achieved for e-learning systems. The overall goal of this thesis was to enable and support modular and multi-granularity reuse of learning resources with a special focus on heterogeneous systems; for compatibility reasons, solutions in particular adopt the SCORM specification as their exchange format.

The overall goal has been pursued through five individual contributions. These contributions focused on the different requirements of multi-granularity reusability. Technical requirements of multi-granularity reusability (availability, retrievability, and interoperability) and three modular operations (modularization, aggregation, and adaptation) have served as a basis for orientation (see Figure 8.1).

The first contribution an extension of SCORM was introduced to enable the modular reuse of learning resources. This extension allows the content of a SCORM package to be distributed to multiple mod-

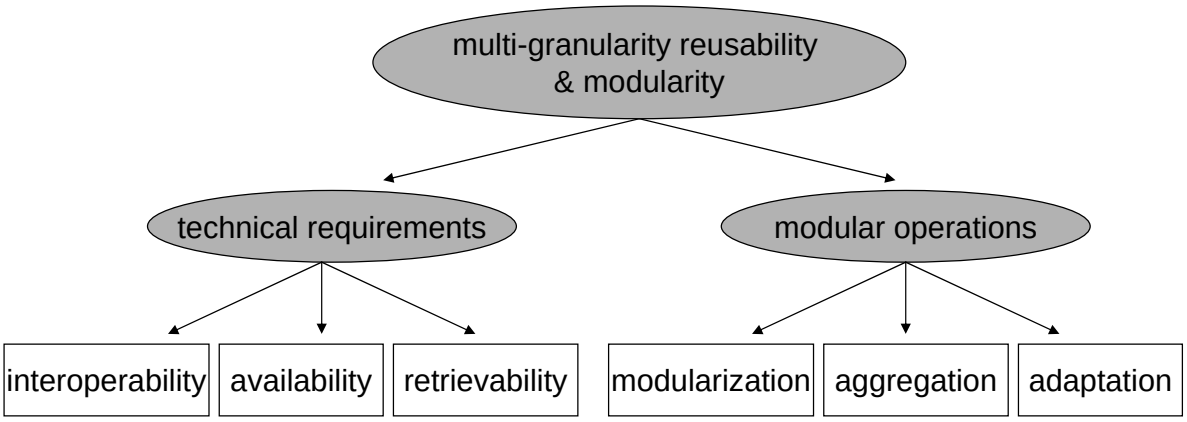


Figure 8.1: Requirements of multi-granularity reusability.

ules. Modules are aggregated by reference; this mechanism does away with the previously required copy&paste operation for reuse.

The second contribution provided a specification of a generic process model for the modularization of learning resources. Assuming that learning resources are often available in non-modular form only, these resources have to be transformed into a modular form; this transformation is called *modularization*. This thesis has analyzed different existing modularization methods. A generic process model of modularization has been created from these various methods. The process model provides a basis for specifying new modularization methods systematically and for comparing different modularization models.

A further contribution is the improvement of the authoring paradigm *authoring by aggregation*. Existing authoring by aggregation systems implement only one of five phases of the ADDIE model[51]. This thesis has extended one authoring by aggregation process by including an additional didactic design phase. Furthermore, the retrieval of learning resources for aggregation has been improved by utilizing knowledge gathered from the current aggregation in progress.

Sometimes it is necessary to adapt learning resources before they can be reused. Adaptation is required particularly if learning resources from different origins are aggregated in order to create a consistent appearance. This thesis contributes a framework for representing and adapting the content of a learning resource. This framework provides an abstraction layer which allows multiple files of a learning resource to be treated as a single logical resource and different document formats to be handled in the same way.

These four contributions have been realized as a prototype as shown in Chapter 7. The prototype serves as a proof-of-concept implementation of the concepts recommended. The prototype demonstrates the feasibility of the concepts: SCORM-based learning resources from different origins can be reused modularly; they can be modularized, aggregated and adapted.

Finally, a new domain-independent categorization approach for learning resources has been developed. This categorization approach can be used to generate topical metadata in order to support the retrieval of learning resources. The advantage of the new approach becomes evident when classifying learning resources from very different domains. Traditional machine learning methods require a training corpus of several manually labeled sample learning resources per category. The new approach this thesis puts forward replaces the conventional sample learning resources by articles from Wikipedia. Therefore, this classifier can be applied even without having any real learning resources available for training. Experiments have shown that accuracy values of 80% may be reached. Although the experiments were conducted with learning resources from one particular project and domain (medical learning resources from the k-MED project) we expect that this method works just as well for text-based learning resources from other domains.

## 8.2 Outlook on Future Research Issues

This thesis has enabled the multi-granularity reusability of learning resources in heterogeneous systems. Concepts which support modular reuse have been proposed and implemented. However, this support is far from perfect. More research issues have to be solved in order to optimize reusability. One important

[51] ADDIE is a commonly accepted model that describes five phases of the whole development of learning resources: analysis, design, development, implementation, and evaluation.

research issue is the adaptation of learning resources. This thesis has provided a framework for the abstraction of learning resource content representation and adaptation. However, how adaptations are performed on a process level was beyond the scope of this thesis. Future research has to provide solutions for reusing learning resources with minimal adaptation effort.

Another research issue that has not been fully resolved is the retrieval of learning resources. The retrieval of learning resources has benefited from advances in the domain of information retrieval; but there are also challenges in learning resource retrieval that differ from typical information retrieval tasks. One such challenge is the ranking of learning resources. The ranking of objects generally depends on a notion of relevance or usefulness of objects to a user's purpose. Relevance and usefulness of learning resources is different from the relevance of documents in Internet search engines. Ochoa et al. have provided the first formal definitions of relevance ranking metrics for learning resources [OD07]. We have proposed another metric that is based on estimations of adaptation effort [ZMRS07]. More research has to be performed in the area of learning resource retrieval in order to improve reusability of learning resources. In particular, the relationship between different learning resources could be of interest.

Automatic metadata generation is yet another topic that is related to the retrieval of learning resources. This thesis has suggested a new approach for the categorization of learning resources by using Wikipedia articles as training corpus. In the scope of this thesis, the method has taken the form of a single-label classifier. For some scenarios of practical application, a multi-label classifier might be more suitable. Further experiments should evaluate how the method may be modified for a multi-label classifier. Furthermore, Wikipedia articles provide more information which can be used to improve the Wikipedia-based classifier. For instance, the use of links between articles, embedded images and markup seems promising.

## Bibliography


[Adv] Advanced Distributed Learning. Sharable Content Object Reference Model (SCORM) 2004 3rd edition.

[Alo04] G. Alonso. *Web Services: Concepts, Architectures and Applications*. Springer, 2004.

[BC99] C.Y. Baldwin and K.B. Clark. *Design rules: volume 1, the power of modularity*. The MIT Press, 1999.

[Ber05] S. Bergsträßer. Automatisierung der Erstellung von Metadaten. Diploma thesis, Technische Universität Darmstadt, Mar 2005.

[Ber08] R. Berbner. *Dienstgüteunterstützung für Service-orientierte Workflows*. Books on Demand Gmbh, 2008.

[BFRS06] S. Bergsträßer, A. Faatz, C. Rensing, and R. Steinmetz. A semantic content representation supporting re-purposing of learning resources. In *Proceedings of I-KNOW '06*, pages 287–295, 2006.

[BLW99] C. Barrit, D. Lewis, and W. Wieseler. CISCO Systems Reusable Information Object Strategy Version 3.0. White Paper, http://www.cisco.com/warp/public/779/ibs/solutions/learning/whitepapers/el_cisco_rio.pdf [last accessed: 2008-06-29], 1999.

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[BPN03] J. Brase, M. Painter, and W. Nejdl. Completing LOM – how additional axioms increase the utility of learning object metadata. In V. Devedzic, J. M. Spector, D. G. Sampson, and Kinshuk, editors, *Proceedings of the 3rd International Conference on Advanced Learning Technologies*, pages 493–493, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[Bru98] R. Brugger. *Eine statistische Methode zur Erkennung von Dokumentstrukturen*. PhD thesis, Universität Freiburg, May 1998.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Harlow, England, 1999.

[CCD+07] A. Campi, S. Ceri, P. Dolog, E. Duval, S. Guinea, G. Houben, D. Massart, M. Nilsson, S. Ternier, and Z. Xuan. ProLearn Query Language Specification. http://ariadne.cs.kuleuven.be/lomi/index.php/QueryLanguages_v1.0 [last accessed: 2008-01-15], 2007.

[CGRU97] C. Chekuri, M.H. Goldwasser, P. Raghavan, and E. Upfal. Web search using automatic classification. In *Proceedings of the Sixth International Conference on the World Wide Web*, 1997.

[CKO92] B. Curtis, M. I. Kellner, and J. Over. Process modeling. *Communications of the ACM*, 35(9):75–90, 1992.

[CMS01] M.F. Caropreso, S. Matwin, and F. Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. *Text Databases and Document Management: Theory and Practice*, pages 78–102, 2001.

[DBD06] B. Doan, Y. Bourda, and V. Dumitrascu. A semi-automatic tool using ontology to extract learning objects. In *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies*, pages 92–93. IEEE Computer Society, 2006.

[DC00] S. Dumais and H. Chen. Hierarchical classification of web content. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval*, pages 256–263, New York, NY, USA, 2000. ACM Press.

[DFC+01] E. Duval, E. Forte, K. Cardinaels, B. Verhoeven, R. Van Durm, K. Hendrikx, M. Wentland Forte, N. Ebel, M. Macowicz, K. Warkentyne, and F. Haenni. The Ariadne knowledge pool system. *Communications of the ACM*, 44(5):72–78, 2001.

[DGJ+04] M. Doorten, B. Giesbers, J. Janssen, J. Daniels, and R. Koper. Transforming existing content into reusable learning objects. *Online education using learning objects. London: Routledge/-Falmer*, pages 116–1127, 2004.

[DH03] E. Duval and W. Hodgins. A LOM research agenda. In *Proceedings of WWW2003-Twelfth International World Wide Web Conference*, pages 20–24, 2003.

[Dit00] J. Dittmann. *Digitale Wasserzeichen: Grundlagen, Verfahren, Anwendungsgebiete*. Springer, 2000.

[DSNK08] P. Dolog, B. Simon, W. Nejdl, and T. Klobučar. Personalizing access to learning networks. *ACM Transactions on Internet Technology*, 8(2):1–21, 2008.

[DV07] D. Dahl and G. Vossen. Lernobjekt-Metadatenerstellung in Zeiten des Web 2.0. *i-com*, 6(2):31–38, August 2007.

[ES01] A. El Saddik. *Interactive Multimedia Learning: Shared Reusable Visualization-based Modules*. Springer-Verlag, 2001.

[ESFS01] A. El Saddik, S. Fischer, and R. Steinmetz. Reusability and adaptability of interactive resources in Web-based educational systems. *ACM Journal of Educational Resources in Computing*, 1(1), 2001.

[Faa04] A. Faatz. *Ein Verfahren zur Anreicherung fachgebietsspezifischer Ontologien durch Begriffsvorschläge*. PhD thesis, Technische Universität Darmstadt, November 2004.

[FFK92] H. Fanderl, K. Fischer, and J. Kaemper. The Open Document Architecture: From Standardization to the Market. *IBM Systems Journal*, 31(4):728–754, 1992.

[FH92] P. H. Feiler and W. S. Humphrey. Software Process Development and Enactment: Concepts and Definitions. Technical report, DTIC Research Report ADA258465, 1992.

[FMMF05] E. Fernandes, H. Madhour, S. Miniaoui, and Maia Wentland Forte. Phoenix tool: A support to semantic learning model. In *Proceedings of the Fifth IEEE International Conference on Advanced Learning Technologies, 2005.*, pages 948–949, 2005.

[FS97] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[GÖ5] M. Görtz. *Effiziente Echtzeitkommunikationsdienste durch Einbeziehung von Kontexten*. PhD thesis, Technische Universität Darmstadt, 2005.

[GJS03] N. Goharian, A. Jain, and Q. Sun. Comparative analysis of sparse matrix algorithms for information retrieval. *Journal of Systemics, Cybernetics and Informatics*, 1(1):38–46, 2003.

[GM07a] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of The 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[GM07b] I. Gurevych and M. Mühlhäuser. Natural Language Processing for Ambient Intelligence . *Special Issue of KI-Zeitschrift ?Ambient Intelligence und Künstliche Intelligenz?*, pages 10–16, 2007.

[Gol90] C. F. Goldfarb. *The SGML Handbook*. Oxford University Press, USA, 1990.

[GSC06] J. Greenberg, K. Spurgin, and A. Crystal. Functionalities for automatic metadata generation applications: a survey of metadata experts' opinions. *International Journal of Metadata, Semantics and Ontologies*, 1(1):3–20, 2006.

[Ham04] S. Hambach. Aus- und Weiterbildung unter besonderer Berücksichtigung von e-Learning : Teil 1: Referenzmodell für Qualitätsmanagement und Qualitätssicherung - Planung, Entwicklung, Durchführung und Evaluation von Bildungsprozessen und Bildungsangeboten. Berlin: Beuth, 2004, 84 pp., Norm Nr.: PAS 1032-1:2004, 2004.

[Han07] A. Hannappel. Anwendung von Verfahren des Maschinellen Lernens für die Klassifikation von Informationsobjekten. Diploma thesis, Technische Universität Darmstadt, 2007.

[HBEV04] P. Haase, J. Broekstra, A. Eberhart, and R. Volz. A Comparison of RDF Query Languages. In *Proceedings of the Third International Semantic Web Conference, Hiroshima, Japan*. Springer, 2004.

[HCC04] C. Huang, S. Chuang, and L. Chien. Liveclassifier: creating hierarchical text classifiers through web corpora. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 184–192, New York, NY, USA, 2004. ACM.

[Hea94] M. A. Hearst. Multi-paragraph Segmentation of Expository Texts. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 9–16, 1994.

[Hea95] M. A. Hearst. Tilebars: visualization of term distribution information in full text information access. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 59–66, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[HHRS05] S. Hoermann, T. Hildebrandt, C. Rensing, and R. Steinmetz. ResourceCenter - A Digital Learning Object Repository with an Integrated Authoring Tool Set. In *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications ED-MEDIA 2005*, pages 3453–3460, Montreal, June 2005. AACE.

[Hoa69] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969.

[Hod02a] H. W. Hodgins. The future of learning objects. In Jack R. Lohmann and Michael L. Corradini, editors, *e-Technologies in Engineering Education: Learning Outcomes Providing Future Possibilities*, volume P1 of *ECI Symposium Series*, pages 281–298, 2002.

[Hod02b] W. Hodgins. IEEE 1484.12.1-2002, Draft Standard for Learning Object Metadata. http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf [last accessed: 2008-06-30], July 2002.

[Hoe05] S. Hoermann. *Wiederverwendung von digitalen Lernobjekten in einem auf Aggregation basierenden Autorenprozess*. PhD thesis, TU Darmstadt, 2005.

[HSW04] J. Haake, G. Schwabe, and M. Wessner. *CSCL-Kompendium: Lehr-und Handbuch zum computerunterstützten kooperativen Lernen*. Oldenbourg Wissenschaftsverlag, 2004.

[HTZH07] C. Huang, Y. Tian, Z. Zhou, and T. Huang. Towards multi-granularity multi-facet e-book retrieval. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 1331–1332, New York, NY, USA, 2007. ACM.

[HWJ94] A. Hampapur, T. Weymouth, and R. Jain. Digital video segmentation. In *MULTIMEDIA '94: Proceedings of the second ACM international conference on Multimedia*, pages 357–364, New York, NY, USA, 1994. ACM.

[IMS01] IMS Global Learning Consortium, Inc. IMS Content Packaging Best Practice Guide - Version 1.1.2 Final Specification. Published online: http://www.imsglobal.org/content/packaging/ cpv1p1p2/imscp_bestv1p1p2.html [last accessed: 2008-02-11], 2001.

[IRC03] A. Ip, A. Radford, and E. Canale. Overcoming the Presentation Mosaic Effect of Multi-Use Sharable Content Objects. In *Proceedings of the 20th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (ASCILITE)*, volume 1, pages 256–262, 2003.

[JC97] J.J. Jiang and D.W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Conference on Research on Computational Linguistics*, pages 19–33, 1997.

[KdHWA04] S. Kabel, R. de Hoog, B. Wielinga, and A. Anjewierden. Indexing Learning Objects: Vocabularies and Empirical Investigation of Consistency. *Journal of Educational Multimedia and Hypermedia*, 13(4):405–425, 2004.

[KE07] S. Kopf and W. Effelsberg. New Teaching and Learning Technologies for Interactive Lectures. *Advanced Technology for Learning Journal*, pages 60–67, 2007.

[KKK06] R. Khoury, F. Karray, and M. Kamel. Extracting and representing actions in text using possibility theory. In *Proceedings of the 3rd annual e-learning conference on Intelligent Interactive Learning Object Repositories (i2LOR 2006)*, 2006.

[KOA03] R. Koper, B. Olivier, and T. Anderson. IMS Learning Design Information Model Version 1.0. http://www.imsglobal.org/learningdesign/ldv1p0/imsld_infov1p0.html [last accessed: 2008-06-20], 2003.

[Kop03] R. Koper. *Reusing Online Resources: A Sustainable Approach to E-learning*, chapter Combining Reusable Learning Resources and Services to Pedagogical Purposeful Units of Learning, pages 46–59. Routledge, 2003.

[Kru01a] U. Kruschwitz. Exploiting structure for intelligent web search. In *34th Annual Hawaii International Conference on System Sciences (HICSS-34)*, volume 4, page 9 pp. IEEE Computer Society, 2001.

[Kru01b] U. Kruschwitz. A rapidly acquired domain model derived from markup structure. In *Proceedings of the ESSLLI'01 Workshop on Semantic Knowledge Acquisition and Categorisation*, 2001.

[KS01] V. Kashyap and L. Shklar. Declarative RDF Models for feature-based targeting of content to multiple devices. In *Proceedings of the Tenth International World Wide Web Conference*, Mar 2001.

[KY07] A. Kolcz and W. Yih. Site-independent template-block detection. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007.*, volume 4702/2007, pages 152–163. Springer Berlin / Heidelberg, 2007.

[LB07] Y. Li and K. Bontcheva. Hierarchical, perceptron-like learning for ontology-based information extraction. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 777–786, New York, NY, USA, 2007. ACM Press.

[LCF+05] E. D. Liddy, J. Chen, C. M. Finneran, A. R. Diekema, S. C. Harwell, and O. Yilmazel. Generating and evaluating automatic metadata for educational resources. In *European Conference on Digital Libraries, Vienna*, pages 513–514. Springer, 2005.

[LGFM08] R. Lokaiczyk, E. Godehardt, A. Faatz, and M. Meyer. On resource acquisition in adaptive workplace-embedded e-learning environments. In *Proceedings of the First International Conference on E-Learning in the Workplace, ICELW2008*, 2008.

[LHRS07] L. Lehmann, T. Hildebrandt, C. Rensing, and R. Steinmetz. Capturing, Management and Utilization of Lifecycle Information for Learning Resources. In *Proceedings of the Second European Conference on Technology Enhanced Learning (EC-TEL 2007)*, volume 4753. Springer, 2007.

[Lie01] M. Liepert. *Rechte, Benutzerrollen und Inhaltsversionierung für verteilte Multimedia-Autorensysteme*. PhD thesis, Technische Universität Darmstadt, Dec 2001.

[Lim04] H. S. Lim. Improving kNN Based Text Classification with Well Estimated Parameters. In *Proceedings of the 11th International Conference on Neural Information Processing, ICONIP 2004, Calcutta, India.*, pages 516–523. Springer, 2004.

[Lin98] D. Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, pages 296–304, 1998.

[LLZ01] L. Lu, S.Z. Li, and H.J. Zhang. Content-based audio segmentation using support vector machines. In *IEEE International Conference on Multimedia and Expo (ICME) 2001*, pages 749–752, 2001.

[LMS05] P. J. Leach, M. Mealling, and R. Salz. A Universally Unique IDentifier (UUID) URN Namespace. Internet proposed standard RFC 4122, July 2005.

[LYRL04] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[Man98] K. Manske. Video browsing using 3D video content trees. In *Proceedings of the 1998 workshop on New paradigms in information visualization and manipulation*, pages 20–24. ACM Press New York, NY, USA, 1998.

[MDO07] M. Meire, E. Duval, and X. Ochoa. SAmgI: Automatic Metadata Generation v2. 0. In *Proceedings of ED-MEDIA 2007, World Conference on Educational Multimedia, Hypermedia & Telecommunications*, 2007.

[Med00] N. Meder. Didaktische Ontologien. In *Globalisierung und Wissensorganisation: Neue Aspekte für Wissen, Wissenschaft und Informationssysteme*, volume 6, pages 401–416. Ergon-Verlag, Würzburg, 2000.

[Med06] N. Meder. *Web-Didaktik*. Bertelsmann, 2006.

[Mey97] B. Meyer. *Object-oriented software construction*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1997.

[MH91] J. Morris and G. Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48, 1991.

[MI04] Y. Matsuo and M. Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(1):157–169, 2004.

[Mil95] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[Mit97] T.M. Mitchell. *Machine Learning*. McGraw-Hill Higher Education, 1997.

[MM04] F. Manola and E. Miller. RDF primer. World Wide Web Consortium, Recommendation REC-rdf-primer-20040210, February 2004.

[MN03] H. Meinedo and J. Neto. Audio segmentation, classification and clustering in a broadcast news task. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03)*, volume 2, 2003.

[MO06] J. Marsh and D. Orchard. XML Inclusions (XInclude) Version 1.0 (Second Edition) - W3C Recommendation 15 November 2006. http://www.w3.org/TR/2006/REC-xinclude-20061115/ [last accessed: 2008-06-30], 2006.

[Mol03] M. Molenda. In search of the elusive ADDIE model. *Performance Improvement*, 42(5):34–36, 2003.

[MPL06] D. Maynard, W. Peters, and Y. Li. Metrics for evaluation of ontology-based information extraction. In *WWW 2006 Workshop on Evaluation of Ontologies for the Web (EON), Edinburgh, Scotland*, 2006.

[MRS07] M. Meyer, C. Rensing, and R. Steinmetz. Categorizing Learning Objects Based On Wikipedia as Substitute Corpus. In Frans Van Assche David Massart, Jean-Noël Colin, editor, *Proceedings of the First International Workshop on Learning Object Discovery & Exchange (LODE'07)*, volume 311, pages 64–71. CEUR Workshop Proceedings, Sep 2007.

[MRS08] M. Meyer, C. Rensing, and R. Steinmetz. Using Community-Generated Contents as a Substitute Corpus for Metadata Generation. *International Journal of Advanced Media and Communication (IJAMC)*, 2(1):59–72, Jan 2008.

[MSB98] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *Research and Development in Information Retrieval*, pages 206–214, 1998.

[MT02] M. Mühlhäuser and C. Trompler. Digital Lecture Halls Keep Teachers in the Mood and Learners in the Loop. In *Proceedings of E-Learn 2002,Montreal, Canada*, pages 714–721. AACE, 2002.

[MT04] A. U. Mauthe and P. Thomas. *Professional Content Management Systems: Handling Digital Media Assets*. John Wiley & Sons, 2004.

[MT07] T. Mikkonen and A. Taivalsaari. Web applications - spaghetti code for the 21st century. Technical Report SMLI TR-2007-166, Sun Microsystems, June 2007.

[NCSS06] D. Newman, C. Chemudugunta, P. Smyth, and M. Steyvers. Analyzing entities and topics in news articles using statistical topic models. In Sharad Mehrotra, Daniel Dajun Zeng, Hsinchun Chen, Bhavani M. Thuraisingham, and Fei-Yue Wang, editors, *Proceedings of the IEEE International Conference on Intelligence and Security Informatics, ISI 2006*, volume 3975 of *Lecture Notes in Computer Science*, pages 93–104. Springer, 2006.

[ND02] F. Neven and E. Duval. Reusable learning objects: a survey of LOM-based repositories. In *MULTIMEDIA '02: Proceedings of the tenth ACM international conference on Multimedia*, pages 291–294, New York, NY, USA, 2002. ACM Press.

[NHHM+04] H.M. Niegemann, S. Hessel, D. Hochscheid-Mauel, K. Aslanski, M. Deimann, and G. Kreuzberger. *Kompendium E-learning*. Springer, 2004.

[Nou05] P. P. Noufal. Metadata: Automatic generation and extraction. In *7th MANLIBNET Annual National Convention on Digital Libraries in Knowledge Management: Opportunities for Management Libraries*, pages 319–327. Indian Institute of Management Kozhikode, May 2005.

[NTD03] J. Najjar, S. Ternier, and E. Duval. The Actual Use of Metadata in ARIADNE: An Empirical Analysis. In *Proceedings of the 3rd Annual ARIADNE Conference*, pages 1–6, 2003.

[NWQ+02] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér, and T. Risch. EDUTELLA: a P2P networking infrastructure based on RDF. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 604–615, New York, NY, USA, 2002. ACM.

[OD07] X. Ochoa and E. Duval. Relevance ranking metrics for learning objects. In *Proceedings of the Second European Conference on Technology Enhanced Learning (EC-TEL 2007)*, volume 4753, pages 262–276. Springer, 2007.

[PC97] J. M. Ponte and W. B. Croft. Text segmentation by topic. In *Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries*, pages 113–125. Springer, 1997.

[Pol03] P. Polsani. Use and abuse of reusable learning objects. *Journal of Digital Information*, 3(4), 2003.

[PS07] S. P. Ponzetto and M. Strube. Deriving a Large Scale Taxonomy from Wikipedia. In *Proceedings of the 22nd Conference on the Advancement of Artificial Intelligence, Vancouver, B.C., Canada, 22-26 July 2007*, pages 1440–1445, 2007.

[PSA08] M. Potthast, B. Stein, and M. Anderka. A Wikipedia-based Multilingual Retrieval Model. In *Proceedings of the 30th European Conference On Information Retrieval*, pages 522–530. Springer, 2008.

[RBH+05] C. Rensing, S. Bergsträßer, T. Hildebrandt, M. Meyer, B. Zimmermann, A. Faatz, L. Lehmann, and R. Steinmetz. Re-Use and Re-Authoring of Learning Resources – Definitions and Examples. Technical Report KOM-TR-2005-02, Technische Universität Darmstadt – Multimedia Communications Lab, November 2005.

[RDL05] D.R. Rehak, P. Dodds, and L. Lannom. A Model and Infrastructure for Federated Learning Content Repositories. In *Interoperability of Web-Based Educational Systems Workshop*, volume 143, 2005.

[Rei83] C.M. Reigeluth. *Instructional-design Theories and Models: An Overview of Their Current Status*. Lawrence Erlbaum Associates, 1983.

[Res95] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, 1995.

[RL07] O. Rostanin and M. Ludwar. From informal learner to active content provider: SLEAM approach. In *Proceedings of the EC-TEL 2007 Poster Session, Crete, Greece, September 17-20, 2007*, 2007.

[RN06] A. Rensink and R. Nederpel. Graph Transformation Semantics for a QVT Language. In *Proceedings of the Fifth International Workshop on Graph Transformation and Visual Modeling Techniques*, pages 45–56, 2006.

[RZM+08] C. Rensing, B. Zimmermann, M. Meyer, L. Lehmann, and R. Steinmetz. Wiederverwendung von multimedialen Lernressourcen im Re-Purposing und Authoring by Aggregation. In Pavlina Chikova Peter Loos, Volker Zimmermann, editor, *Prozessorientiertes Authoring Management: Methoden, Werkzeuge und Anwendungsbeispiele für die Erstellung von Lerninhalten*, pages 19–40. Logos Verlag, Berlin, Jan 2008.

[Sah05] M. Sahlgren. An introduction to random indexing. In *Proceedings of the Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005*, 2005.

[San94] M. Sanderson. Word sense disambiguation and information retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 142–151, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[Sch05] I. Schmitt. *Ähnlichkeitssuche in Multimedia-Datenbanken – Retrieval, Suchalgorithmen und Anfragebehandlung*. Oldenbourg Wissenschaftsverlag, 2005.

[Sch06] P. Schonhofen. Identifying document topics using the wikipedia category network. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 456–462, Washington, DC, USA, 2006. IEEE Computer Society.

[SDHH98] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*, volume 62. Madison, Wisconsin: AAAI Technical Report WS-98-05, 1998.

[Seb02] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Survey*, 34(1):1–47, 2002.

[Seb05] F. Sebastiani. Text categorization. In Alessandro Zanasi, editor, *Text Mining and its Applications to Intelligence, CRM and Knowledge Management*, pages 109–129. WIT Press, Southampton, UK, 2005.

[See02] C. Seeberg. *Life Long Learning: Modulare Wissensbasen für elektronische Lernumgebungen*. Springer, 2002.

[SF02] C. Süß and B. Freitag. LMML–The Learning Material Markup Language Framework. In *Proceedings of the International Workshop Interactive Computer Aided Learning, Villach, Austria*, 2002.

[SFJ07] Z. Syed, T. Finin, and A. Joshi. Wikipedia as an ontology for describing documents. In *Proceedings of the Second International Conference on Weblogs and Social Media*. AAAI Press, 2007.

[SFK00] E. Stamatatos, N. Fakotakis, and G. Kokkinakis. Text genre detection using common word frequencies. In *18th International Conference on computational Linguistics*, pages 808 – 814, 2000.

[SGP+05] M. A. Sicilia, E. Garcia, C. Pages, J. J. Martinez, and J. M. Gutierrez. Complete metadata records in learning object repositories: some evidence and requirements. *International Journal of Learning Technology*, 1:411–424, May 31 2005.

[SKW07] F.M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM Press New York, NY, USA, 2007.

[SL01] A. Sun and E. Lim. Hierarchical text classification and evaluation. In *First IEEE International Conference on Data Mining (ICDM'01)*, page 521, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

[SLN03] A. Sun, E. Lim, and W. Ng. Performance measurement framework for hierarchical text classification. *Journal of the American Society for Information Science and Technology*, 54(11):1014–1028, 2003.

[SMvA+05] B. Simon, D. Massart, F. van Assche, S. Ternier, E. Duval, S. Brantner, D. Olmedilla, and Z. Miklos. A Simple Query Interface for Interoperable Learning Repositories. In *Proceedings of the 1st Workshop on Interoperability of Web-based Educational Systems*, pages 11–18, Chiba, Japan, May 2005. CEUR.

[SN04] R. Steinmetz and K. Nahrstedt. *Multimedia Applications*. Springer, 2004.

[SP06] M. Strube and S.P. Ponzetto. WikiRelate! Computing semantic relatedness using Wikipedia. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence, Boston, MA*, 2006.

[SRS06] P. S. Saini, M. Ronchetti, and D. Sona. Automatic generation of metadata for learning objects. In *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies*, pages 275–279. IEEE Computer Society, 2006.

[SS03a] U. Schroeder and C. Spannagel. Implementierung von eLearning-Szenarien nach der Theorie der kognitiven Lehre. In *DeLFI 2003, Tagungsband der 1. e-Learning Fachtagung Informatik*, 2003.

[SS03b] M. A. Sicilia and S. Sánchez. On the concept of learning object design by contract. *WSEAS Transactions on Computers*, 2(3):612–617, 2003.

[SS04] S. Sanchez and M. A. Sicilia. On the semantics of aggregation and generalization in learning object contracts. In *Proceedings of the 4th IEEE International Conference on Advanced Learning Technologies (ICALT '04)*, 2004.

[SSH05] I. Schmitt, N. Schulz, and T. Herstel. WS-QBE: A QBE-Like Query Language for Complex Multimedia Queries. In *Proceedings of the 11th International Multimedia Modelling Conference*, pages 222–229, 2005.

[Ste99] A. Steinmetz. *Integrierte rechnerbasierte Film- und Filmmetadaten-Präsentationen - Erstellung einer Taxonomie und Entwicklung neuartiger Darstellungs- und Interaktionsformen*. PhD thesis, FB Informatik, TU-Darmstadt, Juli 1999. GMD Research Series No 20/1999; 178 Seiten; GMD Sankt Augustin; 1999; ISSN 1435-2699, ISBN 3-88457-369-1.

[Ste00] R. Steinmetz. *Multimedia-Technologie: Grundlagen, Komponenten und Systeme*. Springer, 2000.

[Ste01] A. Steinacker. *Medienbausteine für web-basierte Lernsysteme*. PhD thesis, Technische Universität Darmstadt, 2001.

[Sum98] K. Summers. Automatic discovery of logical document structure. Technical Report TR98-1698, Cornell University, Computer Science, Aug 1998.

[SvH01] H. Stuckenschmidt and F. van Harmelen. Ontology-based metadata generation from semi-structured information. In *K-CAP '01: Proceedings of the 1st International Conference on Knowledge Capture*, pages 163–170, New York, NY, USA, 2001. ACM Press.

[SVH04] N. Seco, T. Veale, and J. Hayes. An Intrinsic Information Content Metric for Semantic Similarity in WordNet. In *Proceedings of ECAI 2004, the 16th European Conference on Artificial Intelligence. Valencia, Spain.*, volume 4, pages 1089–1090, 2004.

[SWY75] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[TA03] W. Theilmann and M. Altenhofen. Versioning of E-Learning Objects Enabling Flexible Reuse. In *Proceedings of the IADIS International Conference WWW/Internet (ICWI) 2003, Algarve, Portugal, November 5-8, 2003*, pages 719–727, 2003.

[TDV02] S. Ternier, E. Duval, and P. Vandepitte. LOMster: Peer-to-peer Learning Object Metadata. In *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2002*, pages 1942–1943. AACE, 2002.

[The] The Content Sharing project. Project website. http://www.contentsharing.com [last accessed 2008-03-04].

[TMC+08] S. Ternier, D. Massart, A. Campi, S. Guinea, S. Ceri, and E. Duval. Interoperability for Searching Learning Object Repositories. *D-Lib Magazine*, 14(1/2), 2008.

[Tur00] P.D. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4):303–336, 2000.

[VD04] K. Verbert and E. Duval. Towards a global component architecture for learning objects: A comparative analysis of learning object content models. In *Proceedings of the EDMEDIA 2004 World Conference on Educational Multimedia, Hypermedia and Telecommunications*, pages 202–208, 2004.

[VGJD05] K. Verbert, D. Gasevic, J. Jovanovic, and E. Duval. Ontology-based learning content repurposing. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1140–1141, New York, NY, USA, 2005. ACM Press.

[VJD+06] K. Verbert, J. Jovanovic, E. Duval, D. Gasevic, and M. Meire. Ontology-Based Learning Content Repurposing: The ALOCoM Framework. *International Journal on E-Learning*, 5(1):67–74, January 2006. Special Issue: Learning Objects in Context.

[Vos05] J. Voss. Measuring Wikipedia. In *Proceedings 10th International Conference of the International Society for Scientometrics and Informetrics*, 2005.

[Vos06] J. Voss. Collaborative thesaurus tagging the Wikipedia way. http://arxiv.org/pdf/cs.IR/0604036v1 [last accessed: 2008-06-30], April 2006.

[Wes05] M. Wessner. *Kontextuelle Kooperation in virtuellen Lernumgebungen*. Eul Verlag, 2005.

[Wil00] D. A. Wiley. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. http://reusability.org/read/chapters/wiley.doc [last accessed: 2008-06-30], 2000.

[Yan99] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1):69–90, 1999.

[YP97] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 412–420, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.

[ZBC+05] V. Zimmermann, K. Bergenthal, P. Chikova, D. Hinz, L. Lehmann, K. Leyking, G. Martin, and C. Rensing. Authoring Management Platform EXPLAIN. A new learning technology

approach for efficient content production integrating authoring tools through a web-based process and service platform. In *Proceedings of the Ariadne ProLearn Workshop*, Dec 2005.

[ZBRS06] B. Zimmermann, S. Bergsträßer, C. Rensing, and R. Steinmetz. A Requirements Analysis of Adaptations of Re-Usable (E-Learning) Content. In *Proceedings of ED-MEDIA 2006*, 2006.

[Zim08] B. Zimmermann. *Pattern-basierte Prozess-Unterstützung am Beispiel von Anpassungsprozessen*. PhD thesis, Technische Universität Darmstadt, 2008.

[ZMRS07] B. Zimmermann, M. Meyer, C. Rensing, and R. Steinmetz. Improving retrieval of reusable learning resources by estimating adaptation effort. In *Proceedings of the First International Workshop on Learning Object Discovery & Exchange (LODE-2007)*, pages 46–53. CEUR Workshop Proceedings, Sep 2007.

[ZRS06] B. Zimmermann, C. Rensing, and R. Steinmetz. Format-übergreifende Anpassungen von elektronischen Lerninhalten. In *Proceedings of the Deutsche e-Learning Fachtagung Informatik (DeLFI) 2006*, 2006.

[ZRS07] B. Zimmermann, C. Rensing, and R. Steinmetz. Ein Werkzeug zur Unterstützung der Anpassung existierender E-Learning Materialien. In Sigrid Schubert Martin Wessner Christian Eibl, Johannes Magenheim, editor, *DeLFI 2007: 5. e-Learning Fachtagung Informatik*, number P-111, pages 293–294, Köllen, Bonn, 2007. GI, Lecture Notes in Informatics (LNI).

## List of Abbreviations

| | |
|---|---|
| ADDIE | Analysis, Design, Development, Implementation, Evaluation |
| ADL | Advanced Distributed Learning Initiative |
| ALOCoM | Abstract Learning Object Content Model |
| API | Application Programming Interface |
| CAM | Content Aggregation Model |
| CO | Content Ontology |
| CSS | Cascading Style Sheets |
| DF | Document Frequency |
| DIN | Deutsches Institut für Normung e.V. |
| HTML | HyperText Markup Language |
| IEEE | Institute of Electrical and Electronics Engineers, Inc. |
| IMS CP | IMS Content Packaging |
| k-MED | Knowledge in Medical Education |
| kNN | k-Nearest-Neighbors |
| LMS | Learning Management System |
| LO | Learning Object |
| LOM | Learning Object Metadata |
| LOR | Learning Object Repository |
| LR | Learning Resource |
| LRCR | Learning Resource Content Representation |
| LTSC | Learning Technology Standards Committee |
| MERLOT | Multimedia Educational Resource for Learning and Online Teaching |
| MTE | Modification Transaction Engine |
| ODA | Open Document Architecture |

| | |
|---|---|
| OOCR | Object-Oriented Content Representation |
| OWL | Web Ontology Language |
| PDF | Portable Document Format |
| PLQL | ProLearn Query Language |
| RDF | Resource Description Framework |
| RIO | Reusable Information Object |
| RLO | Reusable Learning Object |
| RTE | Run-Time Environment |
| SAmgI | Simple Automatic Metadata Generation Interface |
| SCO | Sharable Content Object |
| SCORM | Sharable Content Object Reference Model |
| SCR | Semantic Content Representation |
| SEC | Semantic Enrichment Component |
| SGML | Standard Generalized Markup Language |
| SQI | Simple Query Interface |
| SQL | Structured Query Language |
| SVM | Support Vector Machine |
| TF-IDF | Term Frequency – Inverted Document Frequency |
| UUID | Universally Unique Identifier |
| VSQL | Very Simple Query Language |
| WBT | Web Based Training |
| XML | Extensible Markup Language |

# Part V

# Appendix

# A Pseudocode for Fragment-Based Categorization Methods

In Chapter 6 three variants of the Wikipedia-based classification method are proposed that operate on a fragment level of learning resources. The pseudocode of these variants is listed here.

**Algorithm A.1:** CATEGORIZATIONFRAGCAT($LR$)

```
procedure CLASSIFYFRAGMENT(F)
  transform learning resource fragment F into a vector representation
  for each article A[i] ∈ WIKIPEDIA
    do calculate similarity σ(F,A[i])
  sort articles by decreasing σ
  S ← top K articles with highest σ
  C ← category that is most frequent in S
  CategoryVoting[C] ← CategoryVoting[C] + 1


main
  Frags ← set of all fragments of learning resource LR
  for each F ∈ Frags
    do CLASSIFYFRAGMENT(F)
  CAT ← category with highest CategoryVoting
  return (CAT)
```

**Algorithm A.2:** CATEGORIZATIONFRAGART($LR$)

```
procedure PROCESSFRAGMENT(F)
  transform learning resource fragment F into a vector representation
  for each article A[i] ∈ WIKIPEDIA
    do calculate similarity σ(F,A[i])
  sort articles by decreasing σ
  S ← top K articles with highest σ
  for each A ∈ S
    do for each category label C of A
    do CategoryVoting[C] ← CategoryVoting[C] + 1


main
  Frags ← set of all fragments of learning resource LR
  for each F ∈ Frags
    do PROCESSFRAGMENT(F)
  CAT ← category with highest CategoryVoting
  return (CAT)
```

**Algorithm A.3:** CATEGORIZATIONFRAGMAX($LR$)

```
procedure PROCESSFRAGMENT(F)
  transform learning resource fragment F into a vector representation
  for each article A[i] ∈ WIKIPEDIA
    do calculate similarity σ(F,A[i])
  comment: Store only the highest value for each article over all fragments
  for each A
    do if σ[F,A] > MaxValues[A]
    then MaxValues[A] ← σ[F,A]


main
  for each A
    do MaxValues[A] ← 0.0
  Frags ← set of all fragments of learning resource LR
  for each F ∈ Frags
    do PROCESSFRAGMENT(F)
  sort articles in MaxValues by decreasing value
  M ← top K articles from MaxValues
  CAT ← category that is most frequent in M
  return (CAT)
```

## B Further Measurements of the Wikipedia-Based Classifier

This appendix provides some more diagrams of measurements from the classification experiments.

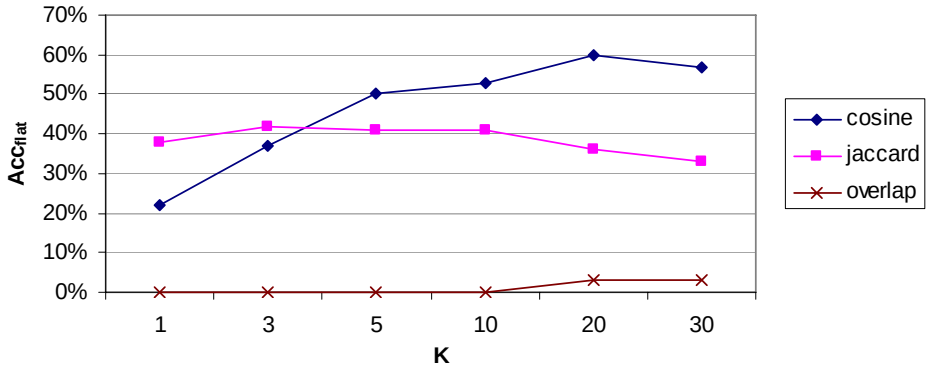


Figure B.1: Comparison of flat accuracy of the three similarity measures for a fixed term selection ($1 \leq DF \leq 50,000$).

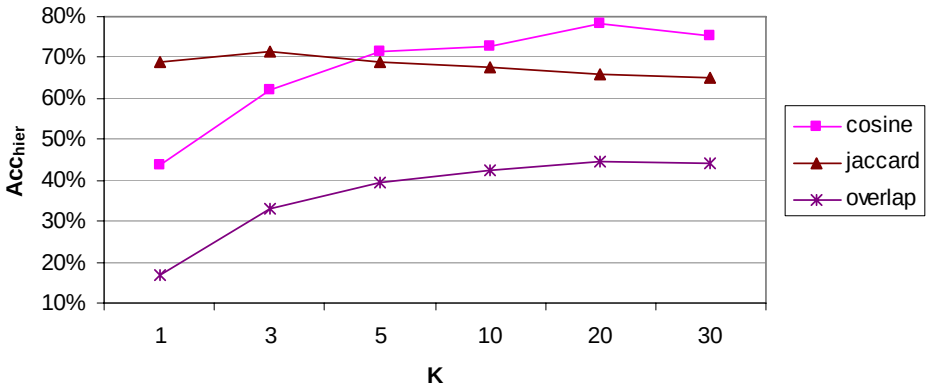


Figure B.2: Comparison of hierarchical accuracy of the three similarity measures for a fixed term selection ($1 \leq DF \leq 50,000$).

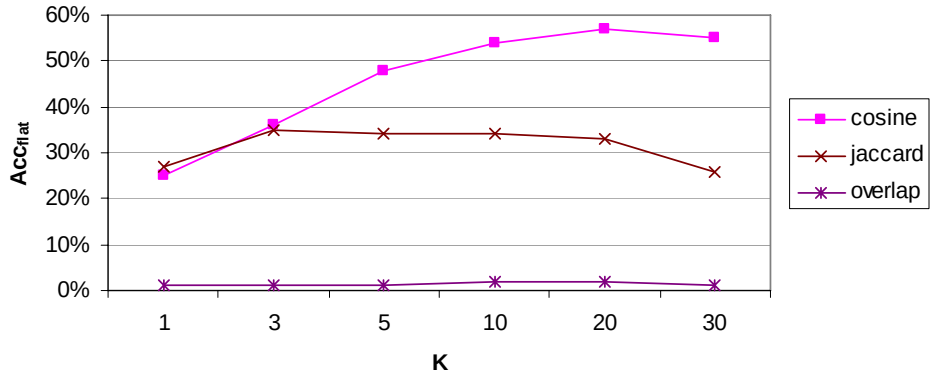


Figure B.3: Comparison of flat accuracy of the three similarity measures without term selection.

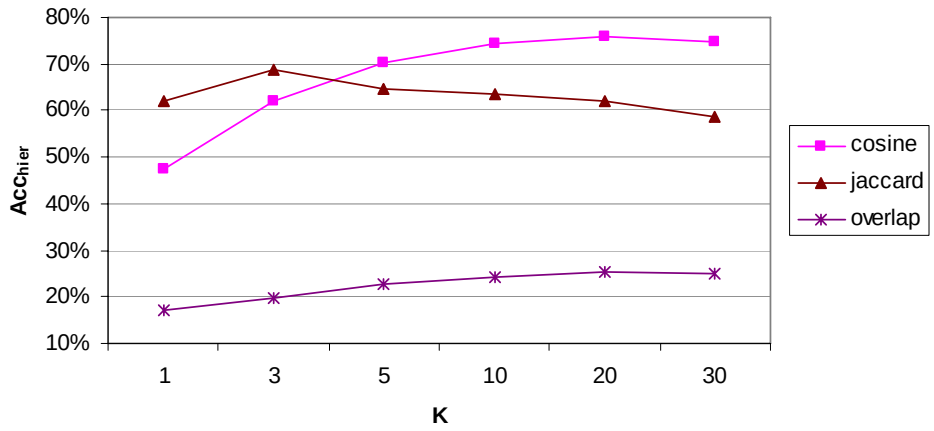


Figure B.4: Comparison of hierarchical accuracy of the three similarity measures without term selection.

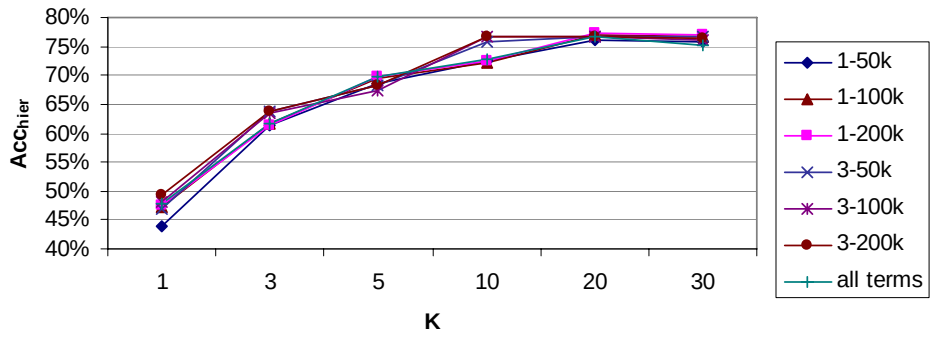


Figure B.5: Comparison of different term selections for hierarchical category propagation (p=0.2).

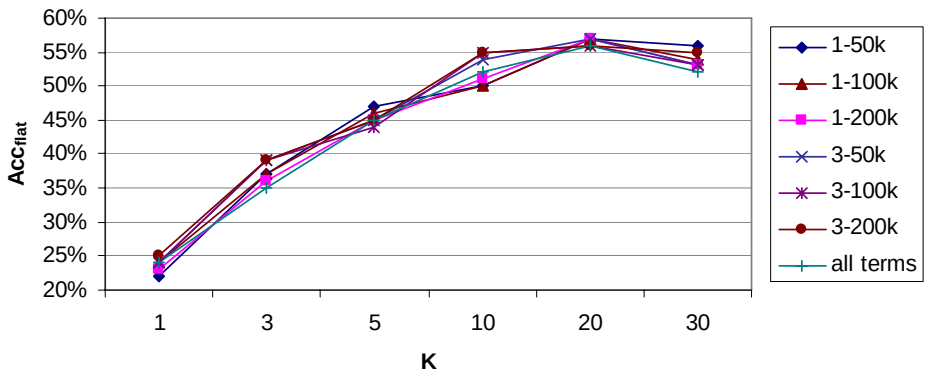


Figure B.6: Comparison of different term selections for hierarchical category propagation (p=0.2).

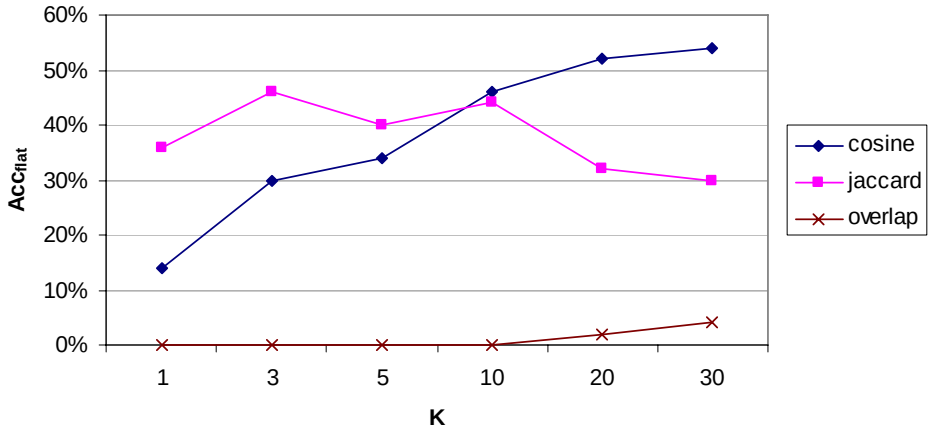


Figure B.7: Flat accuracy measurement for the evaluation set (term selection: $3 \leq DF \leq 100,000$).

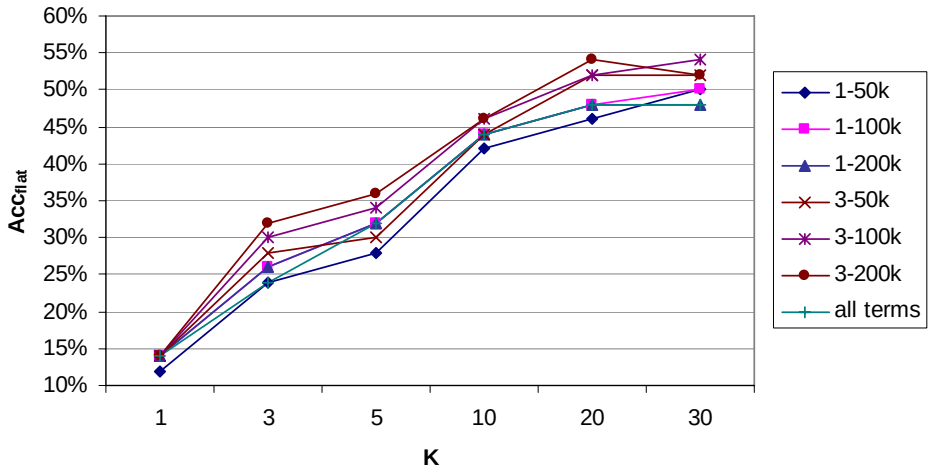


Figure B.8: Comparison of different term selections for the evaluation set with the cosine measure (flat accuracy).

## C Exemplary Module Manifest

An exemplary SCORM manifest is listed below, which shows how the module extension works. The manifest contains inclusions of other modules, and also item references to these manifests.

```
<?xml version="1.0" encoding="UTF-8"?>
  <manifest
    xmlns="http://www.imsproject.org/xsd/imscp_rootv1p1p2"
    xmlns:adlcp="http://www.adlnet.org/xsd/adlcp_rootv1p2"
    xmlns:imsmd="http://ltsc.ieee.org/xsd/LOM"
    xmlns:lom="http://ltsc.ieee.org/xsd/LOM"
    xmlns:xi="http://www.w3.org/2001/XInclude"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" dirty="true"
    identifier="_365870a7-b445-a9bb16343f6c@How_to_organize_a_workshop!"
    xsi:schemaLocation="http://www.imsproject.org/xsd/imscp_rootv1p1p2
␣␣␣␣imscp_rootv1p1p2.xsd␣http://ltsc.ieee.org/xsd/LOM␣schema/lom.xsd
␣␣␣␣http://www.adlnet.org/xsd/adlcp_rootv1p2
␣␣␣␣adlcp_rootv1p2.xsd␣http://www.w3.org/2001/XInclude␣XInclude.xsd">
    <metadata>
      <schema>ADL SCORM</schema>
      <schemaversion>1.2</schemaversion>
        <lom:lom>
          <lom:general>
            <lom:identifier>
              <lom:catalog>http://www.contentsharing.com</lom:catalog>
              <lom:entry>365870a7-b445-a9bb16343f6c</lom:entry>
            </lom:identifier>
            <lom:title>
              <lom:string>How to organize a workshop!</lom:string>
            </lom:title>
          </lom:general>
          <lom:relation>
            <lom:kind>
              <lom:source>http://www.contentsharing.com</lom:source>
              <lom:value>is_revision_of</lom:value>
            </lom:kind>
            <lom:resource>
              <lom:identifier>
                <lom:catalog>http://www.contentsharing.com</lom:catalog>
```

```
        <lom:entry>c3afe017-eae9-222f847f04ae</lom:entry>
      </lom:identifier>
      <lom:description>
        <lom:string>This module is a revision of ...</lom:string>
      </lom:description>
    </lom:resource>
  </lom:relation>
  <lom:relation>
    <lom:kind>
      <lom:source>http://www.contentsharing.com</lom:source>
      <lom:value>include_module</lom:value>
    </lom:kind>
    <lom:resource>
      <lom:identifier>
        <lom:catalog>http://www.contentsharing.com</lom:catalog>
        <lom:entry>3c29e895-6805-9dba47f797bb</lom:entry>
      </lom:identifier>
      <lom:description>
        <lom:string>The referenced module is part has to be
          included for delivery (aggregation by reference).
        </lom:string>
      </lom:description>
    </lom:resource>
  </lom:relation>
  <lom:relation>
    <lom:kind>
      <lom:source>http://www.contentsharing.com</lom:source>
      <lom:value>include_module</lom:value>
    </lom:kind>
    <lom:resource>
      <lom:identifier>
        <lom:catalog>http://www.contentsharing.com</lom:catalog>
        <lom:entry>dcb70b64-69a8-441f98e47e1d</lom:entry>
      </lom:identifier>
      <lom:description>
        <lom:string>The referenced module (...)</lom:string>
      </lom:description>
    </lom:resource>
  </lom:relation>
  <lom:relation> (...) </lom:relation>
</lom:lom>
```

```
    </metadata>
  <organizations default="_365870a7-b445-a9bb16343f6c@TOC1">
    <organization identifier="_365870a7-b445-a9bb16343f6c@TOC1">
      <title>How to organize a workshop</title>
      <item identifier="_365870a7-b445-a9bb16343f6c@ITEM1"
        isvisible="true">
        <title>What is a workshop?</title>
      </item>
      <item identifier="_365870a7-b445-a9bb16343f6c@ITEM2"
        isvisible="true">
        <title>Setting up the workshop</title>
        <item identifier="_365870a7-b445-a9bb16343f6c@ITEM3"
          isvisible="true">
          <title>Defining a topic</title>
        </item>
        <item identifier="_365870a7-b445-a9bb16343f6c@ITEM7"
          isvisible="true">
          <title>Find a suitable location</title>
        </item>
        <item identifier="_365870a7-b445-a9bb16343f6c@ITEM14"
          identifierref="_dcb70b64-69a8-441f98e47e1d@Booking_
␣␣␣␣␣␣␣␣␣␣␣␣Rooms_for_Events"
          isvisible="true">
          <title>Booking Rooms for Events</title>
        </item>
        <item identifier="_365870a7-b445-a9bb16343f6c@ITEM4"
          isvisible="true">
          <title>Find keynote speakers</title>
        </item>
        <item identifier="_365870a7-b445-a9bb16343f6c@ITEM5"
          isvisible="true">
          <title>Invite researchers</title>
        </item>
      </item>
      <item identifier="_365870a7-a614-a9bb16343f6c@ITEM15"
        identifierref="_925ea945-ad5b-5fae323add3c@Publishing_
␣␣␣␣␣␣␣␣␣␣Your_Proceedings">
        <title>Publishing Your Proceedings</title>
      </item>
      <item identifier="_365870a7-b445-a9bb16343f6c@ITEM6"
        isvisible="true">
```

```
          <title>Preparation</title>
          <item identifier="_365870a7-b445-a9bb16343f6c@ITEM9"
            isvisible="true">
            <title>Prepare the location</title>
          </item>
          <item identifier="_365870a7-b445-a9bb16343f6c@ITEM8"
            isvisible="true">
            <title>Care for catering</title>
          </item>
          <item identifier="_365870a7-b445-a9bb16343f6c@ITEM13"
            identifierref="_3c29e895-6805-9dba47f797bb@Catering_
␣␣␣␣␣␣␣␣␣␣␣␣␣for_Events"
            isvisible="true">
            <title>Catering for Events</title>
          </item>
          <item identifier="_365870a7-b445-a9bb16343f6c@ITEM10"
            isvisible="true">
            <title>Find session chairs</title>
          </item>
        </item>
        <item> (...) </item>
      </organization>
    </organizations>
    <resources> (...) </resources>
    <xi:include href="..\3c29e895-6805-9dba47f797bb\imsmanifest.xml"/>
    <xi:include href="..\dcb70b64-69a8-441f98e47e1d\imsmanifest.xml"/>
    <xi:include href="..\76733814-bf5a-71f8c8da7583\imsmanifest.xml"/>
  </manifest>
```

# D List of Own Publications

## D.1 Journals and Book Chapters

1. Marek Meyer, Christoph Rensing, Ralf Steinmetz: Using Community-Generated Contents as a Substitute Corpus for Metadata Generation. In: International Journal of Advanced Media and Communication (IJAMC), vol. 2, no. 1, p. 59–72, January 2008. ISSN 1462-4613.

2. Christoph Rensing, Birgit Zimmermann, Marek Meyer, Lasse Lehmann, Ralf Steinmetz: Wiederverwendung von multimedialen Lernressourcen im Re-Purposing und Authoring by Aggregation. In: Peter Loos, Volker Zimmermann, Pavlina Chikova: Prozessorientiertes Authoring Management: Methoden, Werkzeuge und Anwendungsbeispiele für die Erstellung von Lerninhalten, p. 19–40, Logos Verlag, January 2008. ISBN 978-3-8325-1939-1.

## D.2 Conferences and Workshops

3. Marek Meyer, Christoph Rensing, Ralf Steinmetz: Improving Authoring-by-Aggregation and Using Aggregation Context for Query Expansion. In: E. Duval, R. Klamma, and M. Wolpers: Creating New Learning Experiences on a Global Scale, Second European Conference on Technology Enhanced Learning, EC-TEL 2007, p. 505-510, Springer, September 2007. ISBN 978-3-540-75194-6.

4. Marek Meyer, Christoph Rensing, Ralf Steinmetz: Categorizing Learning Objects Based On Wikipedia as Substitute Corpus. In: David Massart, Jean-Noël Colin, Frans Van Assche: Proceedings of the First International Workshop on Learning Object Discovery & Exchange (LODE'07), vol. 311, p. 64–71, September 2007.

5. Marek Meyer, Alexander Hannappel, Christoph Rensing, Ralf Steinmetz: Automatic Classification of Didactic Functions of e-Learning Resources. In: MULTIMEDIA '07: Proceedings of the 15th international conference on Multimedia, p. 513–516, ACM Press, September 2007. ISBN 978-1-59593-702-5.

6. Marek Meyer, Birgit Zimmermann, Christoph Rensing, Ralf Steinmetz: An Interactive Tool for Supporting Modularization of SCORM-Based Learning Resources. In: Proceedings of ED-MEDIA 2007, p. 3164–3171, AACE, June 2007.

7. Marek Meyer, Sonja Bergsträßer, Birgit Zimmermann, Christoph Rensing, Ralf Steinmetz: Modeling Modifications of Multimedia Learning Resources Using Ontology-Based Representations. In: Tat-Jen Cham and Jianfei Cai and Chitra Dorai and Deepu Rajan and Tat-Seng Chua and Liang-Tien Chia: Advances in Multimedia Modeling, vol. LNCS 4351, p. 34–43, Springer, January 2007. ISBN 978-3-540-69421-2.

8. Marek Meyer, Christoph Rensing, Ralf Steinmetz: Towards Using Wikipedia as a Substitute Corpus for Topic Detection and Metadata Generation in E-Learning. In: Proceedings of the 3rd annual e-learning conference on Intelligent Interactive Learning Object Repositories, November 2006.

9. Marek Meyer, Tomas Hildebrandt, Christoph Rensing, Ralf Steinmetz: Requirements and an Architecture for a Multimedia Content Re-purposing Framework. In: Wolfgang Nejdl and Klaus Tochtermann (Edits.): First European Conference on Technology Enhanced Learning - EC-TEL 2006, p. 500–505, Springer Verlag, October 2006. ISBN 3-540-45777-1.

10. Marek Meyer: Modularization of Existing Learning Resources for Repurposing. In: Katherine Maillet and Ralf Klamma: Proceedings of the 1st Doctoral Consortium in Technology Enhanced Learning, p. 39–44, October 2006.

11. Marek Meyer, Christoph Rensing, Ralf Steinmetz: Modellierung eines generischen Prozesses für die Modularisierung von Lernressourcen. In: Christoph Rensing (Edit.): Proceedings der Pre-Conference Workshops der 4. e-Learning Fachtagung Informatik DeLFI 2006, p. 19–26, Logos, September 2006. ISBN 3-8325-1330-2.

12. Marek Meyer, Christoph Rensing, Ralf Steinmetz: Supporting Modularization and Aggregation of Learning Resources in a SCORM Compliance Mode. In: Kinshuk X.; Koper, R.; Kommers, P.; Kischner, P.; Sampson, D.G.; Didderen, W. (Edits.): Advanced Learning Technologies, 2006. Sixth International Conference on , p. 933 - 935 , IEEE Computer Society Press, July 2006. ISBN 0-7695-2632-2.

13. Robert Lokaiczyk and Eicke Godehardt and Andreas Faatz and Marek Meyer: On Resource Acquisition in Adaptive Workplace-Embedded E-Learning Environments. In: Proceedings of the First International Conference on E-Learning in the Workplace, ICELW2008, June 2008.

14. Birgit Zimmermann, Marek Meyer, Christoph Rensing, Ralf Steinmetz: Improving Retrieval of Reusable Learning Resources by Estimating Adaptation Effort. In: Proceedings of the First International Workshop on Learning Object Discovery & Exchange (LODE-2007), p. 46–53, September 2007.

## D.3 Technical Reports

15. Sonja Bergsträßer, Birgit Zimmermann, Marek Meyer, Christoph Rensing, Andreas Faatz, Tomas Hildebrandt, Ralf Steinmetz: Re-Purposing: Motivation, Related Work, and Building Blocks. no. KOM-TR-2005-03, December 2005.

16. Christoph Rensing, Sonja Bergsträßer, Tomas Hildebrandt, Marek Meyer, Birgit Zimmermann, Andreas Faatz, Lasse Lehmann, Ralf Steinmetz: Re-Use and Re-Authoring of Learning Resources - Definitions and Examples. no. KOM-TR-2005-02, November 2005.

## D.4 Patent Applications

17. Marek Meyer, Tomas Hildebrandt: Hierarchical Metadata Generator for Retrieval Systems. (submitted to USPTO in 2007)

18. Marek Meyer, Andreas Faatz, Tomas Hildebrandt: Generating Searchable Keywords. (submitted to USPTO in 2006)

19. Marek Meyer, Tomas Hildebrandt: Lossless format-dependent analysis and modification of multi-document e-learning resources. (submitted to USPTO in 2005)

20. Birgit Zimmermann, Marek Meyer: Estimation of adaptation effort based on metadata similarity. (submitted to USPTO in 2006)

# E Lebenslauf des Verfassers (Curriculum Vitae)

## Persönliche Daten

| | |
|---|---|
| Name: | Marek Meyer |
| Geburtsdatum: | 03. September 1979 |
| Geburtsort: | Bad Soden am Taunus |
| Nationalität: | Deutsch |

## Ausbildung

| | |
|---|---|
| 1999 – 2005 | Technische Universität Darmstadt<br>Studiengang: Informatik (Diplom)<br>Abschluss: Diplom-Informatiker, Note: mit Auszeichnung bestanden |
| 1996 – 1999 | Friedrich-Dessauer-Gymnasium Frankfurt am Main, Abschluss: Abitur |
| 1990 – 1996 | Leibnizschule Frankfurt am Main |
| 1986 – 1990 | Robinson-Grundschule Hattersheim |

## berufliche Tätigkeit

| | |
|---|---|
| 2005 – heute | Technische Universität Darmstadt, Fachgebiet Multimedia Kommunikation<br>Wissenschaftlicher Mitarbeiter, Forschungsgruppe Knowledge Media |
| 2005 – 2008 | SAP AG, SAP Research CEC Darmstadt<br>Doktorand |
| 2003 – heute | Richter Meyer Media GbR, Mitinhaber<br>Hosting von Web Content Management Systemen (Typo3) |
| 1999 – 2004 | Infraserv GmbH & Co. Höchst KG<br>Werkstudent |
| 1998 | Infraserv GmbH & Co. Höchst KG<br>Ferienjob als Aushilfe |

## Betreute Studien- und Diplomarbeiten

Tobias Ott. Webservice-basierte Suche im e-Learning. Bachelorarbeit, Technische Universität Darmstadt, März 2008.

Stanislava Trambabova. Analysis of Statistical Significance of Similarity Measures for the Determination of Learning Object Granularity. Diplomarbeit, Technische Universität Darmstadt, Juni 2007.

Alexander Hannappel. Anwendung von Verfahren des Maschinellen Lernens für die Klassifikation von Informationsobjekten. Diplomarbeit, Technische Universität Darmstadt, Juni 2007.

Anika Schneider. Supporting Aggregation of Learning Resources by Semantic Metadata. Diplomarbeit, Technische Universität Darmstadt, Juni 2007.

Fouad Korkmaz. Übertragung von Verfahren zur Textsegmentierung auf Lernressourcen. Bachelorarbeit, Technische Universität Darmstadt, Juni 2006.