Chapter 1

Introduction

> *"The reasonable man adapts himself to the world; the unreasonable one persists in trying to adapt the world to himself. Therefore all progress depends on the unreasonable man."*[1]

1.1 Background and Motivation

Since the emergence of the World Wide Web (WWW), the size of its audience and the amount of information published on it have grown enormously. The survey "How much information?", carried out in the end of 2003 at the University of California at Berkeley, reports of an Internet community counting more than 600 million members, 167 Terabytes of published Web content, and prognosticates an annual doubling for the future [Lyman et al. 2003]. Another important trend is the Web's evolution from a simple presentation medium to a dynamic medium of interaction and communication. Whereas the first Web sites consisted of a small collection of HTML pages presenting mainly static content, today's sites act as complex *Web Information Systems (WIS)* that provide both up-to-date information and functionality. It is no exaggeration to claim that the World Wide Web has changed the way of accessing and processing information fundamentally. Meanwhile, it offers electronic services in a number of different application areas, among them the press, education, culture, entertainment, tourism, commerce, administration, etc.

This heterogeneity characterizes not only the application areas of the WWW, but also its growing community. Even though originally developed for the academic field, the Web has quickly found its way to the general public. Today's Web users significantly differ in age, education, preferences, interests, capabilities, cultural background, etc. Furthermore, they access the Web from a growing diversity of locations and client devices. Besides traditional desktop computers and Web browsers, the usage of mobile appliances (e.g. smart phones, PDAs, notebooks, set top boxes) is rapidly increasing, each provided with different presentation, communication, and interaction features.

These trends necessitate the appropriate selection, generation, and delivery of up-to-date information that is automatically adjusted to the user, the capabilities of his client device, and his entire *usage context*. The result is *adaptive Web sites*: sites that are customized (personalized) to better meet the specific requirements, preferences, and characteristics of their audience. Typically, this customization needs to concern different aspects of a Web application: the *data* it presents, the *navigational structure* and it offers on top of this data in terms of interlinked Web pages, and the visual *presentation* of those Web pages. To efficiently cope with this multitude of adaptation issues has become a significant challenge

[1] George Bernard Shaw (1856 - 1950), Man and Superman (1903), Maxims for Revolutionists

for today's Web site providers and developers. It implies additional costs and efforts in the design, implementation, and maintenance of Web information systems.

The development of early Web-based systems was typically characterized by the ad hoc authoring of (mainly static) Web documents. Still, the growing complexity of Web applications has soon made clear that such a straightforward approach is not sufficient when creating and maintaining complex Web sites. The recently emerged *Web engineering* research field tackles this problem by the "establishment and use of sound scientific, engineering and management principles" to Web site development [Murugesan and Deshpande 2001]. Inspired by the principles of traditional software engineering, Web engineering aims at adopting its well-proven methods and concepts to the specific characteristics of Web-based systems. Furthermore, triggered by the aforementioned demand for personalization and device independence, it places an increasing emphasis on considering the specific requirements for engineering adaptive Web sites. This additional consideration of adaptation (*adaptation engineering*) concerns different phases of a Web application's overall life-cycle: its design, implementation, publication, maintenance, testing, evolution, etc. Furthermore, besides the structured development of "new" adaptive Web sites, a research question gaining always more importance is how already existing Web information systems can be extended with additional adaptation functionality.

A basic observation in the Web engineering field was that Web-based applications lacking a systematic underlying design suffer from enormous usability and manageability problems [Murugesan et al. 2001]. Thus, in the last decade, several Web design methods have been proposed by academia (RMM [Isakowitz et al. 1995], OOHDM [Schwabe et al. 1996], WebML [Ceri et al. 2000], WSDM [De Troyer 2001], Hera [Vdovjak et al. 2003], etc.). Their main goal is to simplify the development of Web sites by abstracting from the implementation and separately considering different design aspects. While applying different techniques and notations, a common characteristic of all design methods is to distinguish between a data, a navigation, and a presentation design. Selected methods (e.g. OOHDM, WebML, WSDM, Hera) also offer some support for personalization and adaptation at design-level. However, the provided adaptation is mostly centered around certain content and navigation adaptation aspects. Important context-dependency issues, such as media adaptation or (dynamic) adaptation at presentation design have not been sufficiently addressed, yet. Moreover, there is only limited tool support for the visual specification of adaptation and the (semi-)automatic generation of a corresponding adaptive implementation. Furthermore, no design method facilitates the extension of an already existing Web application with additional adaptation concerns.

Another important and yet not sufficiently addressed shortcoming is the current coarse-grained implementation model of the WWW. While it is well-suited for easy authoring and straightforward publication of documents, it is obviously not a sufficient implementation base for structured Web engineering approaches [Gaedke et al. 2000]. Though some of the aforementioned Web design methods provide a (semi-)automatic implementation generation, fine-granular semantic, navigational, and presentational design elements get lost during the implementation phase while being transformed into (X)HTML, cHTML, WML documents, etc. These document formats have significant disadvantages concerning their applicability for implementing and managing adaptive, dynamic Web information systems. The missing separation of content, layout, and structure prevents to uniformly create, update, and reuse content for different user preferences and platforms [Gellersen et al. 1997]. Furthermore, no mechanisms are provided to describe the adaptive behavior of reusable content pieces (Web code) in a generic way. This lack of structure and configurability in Web application code prevents the reuse of independent, configurable, and adaptable implementation artefacts both

within an application and for other applications and target systems. As a consequence, most Web site providers today use to create and manage Web code for different platforms and usage contexts separately.

At the same time, the efficient reuse of application code is a main task of traditional software technology, and has already been successfully addressed by component-based software engineering (CBSE [Szyperski 1998]). The advantages of component models are numerous: reusability, system-independence, configurability, flexibility, composability, etc. Traditional component models consider components as binary units of composition, mostly based on imperative programming languages. Still, in the recent years different approaches have been proposed to apply their principles to the document-centric nature of Web and multimedia applications [Gellersen et al. 1997, Aßmann 2005]. Meanwhile, there exist a number of structured, declarative, component-based document formats for different application areas, such as hypermedia presentations (HMDoc [Westbomke 2001]), Web applications (WebComposition [Gaedke et al. 2000]), eLearning applications (CHAMELEON [Wehner and Lorz 2001]) or even for Web-based three-dimensional user interfaces (CONTIGRA [Dachselt et al. 2002]). Still, even though all these approaches benefit from the reuse of declarative and configurable implementation artefacts in a component-like manner, none of them provides support for the aforementioned adaptation issues, such as device-independence, personalization, or localization. Moreover, there is a lack of solutions for the automatic generation of a component-based implementation based on high-level Web design specifications.

To fill this gap, the approach proposed in this dissertation focuses on the component-based development of adaptive Web applications. The vision is the intuitive composition of personalized, device-independent Web presentations from declarative, reusable, and adaptive "building blocks" (components), aided by a structured design and development process. In order to fulfill this vision, this thesis proposes a *concern-oriented component model* for adaptive Web applications[2,3]. It is based on the notion of declarative *document components* that encapsulate separate application aspects on different abstraction levels and can be automatically adjusted to varying user preferences and client platforms. For the development of component-based adaptive Web documents a structured *multi-stage, model-based authoring process* is presented. Distinguishing between different phases of design and component-based implementation, it allows component authors to systematically take into account various application and adaptation concerns. The intuitive creation and composition of document components is supported by a visual authoring tool. What more, it is illustrated how a component-based implementation can be automatically generated from a high-level design specification in a model-driven way, allowing to add automation to the overall process of design and implementation. The resulting development process supports different kinds of content, navigation, and presentation layer adaptation, and is demonstrated by a number of representative examples. Finally, it is shown how the lessons learned from authoring component-based adaptive Web applications can be generalized in order to add adaptation to existing (not component-based) Web applications.

As its main benefit, the proposed approach enables Web engineers to efficiently develop adaptive Web applications from reusable components, by systematically taking into account different application and adaptation concerns from design to implementation. According to

[2]The component model presented in this dissertation was developed within the scope of the AMACONT research project [@AMACONT] and is also often referred to as the AMACONT component model. The thesis presents the author's contributions to the model, based on requirements towards the efficient authoring of adaptive Web applications from reusable components.

[3]The term *concern-oriented* denotes the model's support for the clear separation of concerns involved in a Web application, each being dealt with on different component levels.

our best knowledge, it is the first development method for personalized, context-adaptive Web applications that combines the advantages of model-based Web design methodologies (e.g. high-level specification, thorough separation of design concerns, etc.) with the benefits of a component-based implementation techniques (such as reusability, configurability, or self-adaptation). To achieve this overall goal, a number of scientific contributions are provided, among them the design of a novel-component model for adaptive Web applications, its combination with a model-based Web design method, the provision of design-time support for presentation layer adaptation, as well as a means for easily adding adaptation to existing Web presentations.

1.2 Problems, Theses, and Research Goals

After outlining the research context and the main vision of this dissertation, this section recapitulates the concrete problems to be solved and derives research theses and goals resulting from them. First, the main shortcomings to be addressed are comprised:

Problems

- The development of adaptive Web applications requires significantly higher efforts compared to non-adaptive Web-based systems, there are lacking design guidelines and hardly any reuse concepts. The consideration and realization of different adaptation aspects (personalization, device independence, localisation, etc.) is a great challenge in designing, implementing, and maintaining Web sites.
- Current Web document formats are not suitable for developing personalized, device independent Web applications. The coarse-grained implementation model of the WWW prevents the efficient reuse of fine-granular implementation artefacts in a component-wise manner. Furthermore, there is a lack of mechanisms for describing the adaptive behavior of reusable content pieces (Web code) in a generic way.
- Due to their lacking support for adaptation, existing authoring tools for engineering dynamic Web applications are not suitable for personalized, device independent applications
- Existing authoring tools for adaptive hypermedia and Web-based systems are mostly restricted to the abstract level of conceptual modeling, not allowing for the visual creation of reusable adaptive implementation artefacts.
- Current design models for hypermedia and Web applications are only inadequately suitable for the specification of personalized ubiquitous Web applications. They do not provide sufficient support for specifying important concerns such as adaptation at presentation design or dynamic adaptation.
- There is a lack of solutions allowing for translating high-level design artefacts to an adaptive implementation layer in a model-driven way, thus adding automation to the overall development process of dynamic adaptive Web applications.
- Current solutions for engineering adaptive hypermedia and Web-based applications assume their development “from scratch”. There is no sufficient support (neither at design nor at implementation level) for easily adding adaptation to an existing Web application.

Theses

Based by these shortcomings and the situation of this dissertation in the research context, the following list comprises the main research theses behind the proposed approach. These theses also serve as the motivation of this work.

- Declarative component-based XML document formats combined with visual authoring tools are well applicable for engineering Web-based multimedia applications, among them personalized adaptive Web presentations. Still, existing component-based formats do not provide sufficient support for adaptive Web applications.
- The development process of component-based adaptive Web applications has to be based on a structured design and authoring process that is aided by appropriate authoring tools. These have to support a strict separation of concerns as well as varying aspects of adaptation in all different authoring steps.
- The combination of formalized, high-level design models with declarative, component-based document formats allows to automate the overall process of designing and implementing adaptive Web sites in a model-driven way.
- Adaptive Web Information Systems (AWIS) can be reduced to a series of data transformations. Furthermore, major parts of the adaptation-specific data transformations can be clearly separated from the rest of a Web application. Thus, the mechanisms for realizing adaptation in component-based Web applications can be generalized for adding adaptation to existing XML-based Web applications by using generic transcoding tools.
- Based on a declarative rule language, such generic adaptation components can be easily configured to implement various kinds of adaptation in existing Web Information Systems.

Research Goals

Triggered by the outlined shortcomings and theses, the vision of this dissertation is the efficient component-based development of adaptive Web applications in combination with a structured, model-based authoring process as well as visual authoring tools. In order to fulfill this vision and to elaborate the proposed theses, the following research goals have to be accomplished within the scope of this work:

- Design of a novel, XML-based, concern-oriented component model for engineering dynamic adaptive Web applications. Compact introduction and overview of the model's language constructs, its XML-based description language, as well as its selected benefits.
- Design of a structured authoring process for the component-based development adaptive Web sites. Adoption of the model-based Hera design method for engineering data-driven Web Information Systems from declarative document components. Explanation of the resulting Hera-AMACONT[4] methodology based on a running example.
- Design and prototypical implementation of a visual authoring tool called AMACONT-Builder for component-based adaptive Web applications. Compact introduction of its basic concepts and main modules from the author's point of view.

[4]Note that the work described in this dissertation was carried out as part of the AMACONT research project [@AMACONT].

- Design and prototypical implementation of a solution for the model-driven generation of component-based adaptive Web presentations. RDF(S)-based formalization of the presentation design phase of the Hera-AMACONT methodology. Realization of a model-driven transformation architecture for automatically translating high-level design artefacts to a component-based implementation supporting different aspects of static and dynamic adaptation.

- Design and implementation of mechanisms for separating adaptation-specific transformations from adaptive Web applications. Development of the GAC (Generic Adaptation Component), a generic transcoding component for making existing Web applications adaptable and adaptive.

- Specification of an RDF-based declarative GAC configuration language for the application-independent definition of adaptation and context data rules. Implementation of the GAC as well as demonstration of its main functionality by adding adaptation to an existing Web application.

1.3 Outline of the Dissertation

The rest of the dissertation is structured as follows.

Chapter 2 - Adaptive Hypermedia and Web-based Systems

Chapter 2 provides a short introduction to adaptive hypermedia and Web-based applications. Basic definitions are provided as well as main application areas, methods, and techniques of hypermedia adaptation are summarized. Furthermore, existing reference models for adaptive hypermedia and Web applications are described.

Chapter 3 - Development of Adaptive Web Applications: State of the Art

Chapter 3 deals with the development of adaptive Web applications and covers existing work on related Web engineering approaches. A main focus is on component-based and document-oriented approaches aiming at the implementation of multimedia and Web applications from declarative reusable implementation entities. Furthermore, model-based approaches supporting the structured design and development of hypermedia and Web-based systems are also discussed in detail. While examining related work, a special focus is on the question of how adaptation and personalization are supported. Limitations of existing approaches are observed as well as additional requirements for the development of adaptive Web-based systems are identified and discussed.

Chapter 4 - A Concern-Oriented Component Model for Adaptive Web Applications

Chapter 4 presents a concern-oriented component model for dynamic adaptive Web applications. The concept of declarative adaptable document components is introduced, and a corresponding XML-based component description language is presented. The document model enables to encapsulate adaptable content to document components on different levels of abstraction and provides support for describing both their adaptive behavior and adaptive

presentation layout. Furthermore, the concept of dynamic component templates is introduced, allowing to realize data-driven adaptive Web presentations. For the on-the-fly publishing of component-based adaptive Web applications a pipeline-based document generator is presented. Finally, significant model benefits are mentioned.

Chapter 5 - The Authoring Process and its Tool Support

Chapter 5 deals with the authoring process of component-based adaptive Web applications. Different application areas and possible process models are discussed, but the main focus is on the structured development of data-driven adaptive Web presentations. For this purpose the model-based Hera design method [Vdovjak et al. 2003] is adopted and extended to the context of component-based Web engineering, resulting in the so-called Hera-AMACONT methodology. Considering the different design steps identified by Hera-AMACONT as a guideline, it is shown how component authors can systematically create, configure, aggregate, and interlink document components to complex adaptive Web presentations.

As a flexible authoring tool for component developers the AMACONTBuilder is introduced. Based on an extensible set of graphical editor modules, it allows to create adaptive document components (and templates) on different abstraction levels. The tool is shown to facilitate different authoring scenarios, as it is independent of any one specific methodology. Furthermore, selected implementation and extensibility issues are also briefly presented.

While the AMACONTBuilder facilitates flexible component authoring (implementation) independent of a specific design method, in some cases it is desirable to add automation to the overall process of design and implementation. Therefore, Chapter 5.3 deals with the research question of how high-level design specifications can be automatically translated to a component-based adaptive Web presentation in a model-driven way. After identifying main automation requirements, an RDF(S)-based formalization of the presentation design phase of the Hera-AMACONT methodology is proposed. According to this formalization, high-level model specifications can be automatically mapped to a component-based implementation, thus exploiting its flexible presentation and adaptation capabilities. The resulting multi-stage hypermedia generation process is exemplified by a prototype application.

Chapter 6 - A Generic Transcoding Tool for Making Web Applications Adaptive

The combination of the component-based document format with an authoring process relying on a structured design method provides an efficient framework for developing adaptive Web-based systems. Still, the resulting engineering process assumes to build adaptive Web applications "from scratch", not providing support for developers aimed at adding adaptation to already existing applications. Therefore, Chapter 6 addresses the research question of how the lessons learned from developing component-based adaptive Web presentations can be generalized for extending a broader range of Web presentations with additional adaptation functionality. For this purpose a flexible transcoding tool called the Generic Adaptation Component (GAC) is introduced. Configured by an RDF-based rule language, it allows the addition of both static and dynamic adaptation to Web presentations. To prove the concept's feasibility, it is shown how GACs can be configured to add adaptation to an existing Web application.

Chapter 7 - Conclusion and Future Work

Chapter 7 provides a summary of the thesis and outlines its main contributions, achievements, but also its boundaries and limitations. Furthermore, possible extensions as well as targeted future work are discussed.

1.4 Writing Conventions

When reading this dissertation, following issues concerning the writing style should be taken into account:

- Some of the names of companies and/or products mentioned in this dissertation are registered trademarks. They are used without any warranty and are not explicitly marked in the text.
- Whenever a new (technical) term is mentioned or introduced in the thesis, its first occurrence is made stand out by using *italics*. Any further occurrences are not explicitly emphasized.
- Often-used abbreviations are listed and resolved in the Table of Abbreviations at the end of the dissertation.
- All cited literature references can be found in the Bibliography at the end of the thesis. References to Web sites of companies, products, projects, and prototype demonstrators are labeled by an additional @ prefix, e.g. [@AMACONT].
- At the end of the dissertation, there is an Index of the most important terms, concepts, and names.
- Listings and source code snippets are presented with numbered lines with syntax highlighting. Whenever there are deviations between the presented source code examples and their original form (e.g. for the sake of better readability or understandability), these differences are explicitly mentioned. A list of all source examples can be found at the beginning of the dissertation.
- The figures and tables presented in this dissertation were either created by the author, or their origin is explicitly mentioned (e.g. by an explicit remark in the text or the appropriate literature reference to their source). A list of all figures (List of Figures) and tables (List of Tables) can be found at the beginning of the dissertation.

