REPORTS
IN
INFORMATICS

ISSN 0333-3590

Composing Web Presentations using Presentation Patterns

Khalid A. Mughal Yngve Espelid Torill Hamre

REPORT NO 331 August 2006

Department of Informatics
UNIVERSITY OF BERGEN
Bergen, Norway

This report has URL http://www.ii.uib.no/publikasjoner/texrap/pdf/2006-331.pdf

Reports in Informatics from Department of Informatics, University of Bergen, Norway, is available at http://www.ii.uib.no/publikasjoner/texrap/.

Requests for paper copies of this report can be sent to:
Department of Informatics, University of Bergen, Høyteknologisenteret,
P.O. Box 7800, N-5020 Bergen, Norway

Composing Web Presentations using Presentation Patterns[*]

Khalid A. Mughal[†] Yngve Espelid[†] Torill Hamre[‡]

Abstract

This report proposes the concept of a presentation pattern as a mechanism to capture the key aspects of a web presentation: layout, navigation and data structure. We describe the specification of presentation patterns and outline the workflow in a system that uses these patterns to generate web presentations. This system creates online courses based on presentation patterns that we have developed. Features of such an online presentation are demonstrated. Experience from the use of the system is discussed, and new directions for future work are outlined. Our approach promotes reuse of content and presentation patterns for developing online courses.

Keywords: web presentations, presentation patterns, dynamic publishing, decoupling content and formatting, e-Learning

1 Introduction

Consider a scenario where a web page designer, who is satisfied with the web pages that comprise a presentation on the net, would like to *replicate* the web pages with new content. In the worst case, such an endeavour can entail starting from scratch: re-implementing the web pages with the new content. The problem is invariably that the formatting is tightly coupled with the content, and decoupling of the two is not always trivial. Cutting and pasting between existing and new pages is not always a viable option, and it is not certain that the web page designer has any inclination towards programming such a task. It is also not always the case that there is a budget to invest in a full-blown proprietary system for creating web publications. A simple Open Source-based system that could take the new content and create a new presentation based on the formatting and the functionality of the existing web pages would be very welcome.

We are experimenting in generating net-based presentations (for example, online courses) that are based on presentation patterns. A *presentation pattern* specifies the pertinent aspects of a presentation: page rendering, the navigation and the requirements for the content it can display, so that a presentation can be generated by supplying the right kind of data. This strategy decouples the content from the formatting, and both the content and the presentation pattern can be reused. This decoupling means that the same content can be used to create other types of presentations based on different presentation patterns, and the same presentation pattern can be applied to different content to create same type

[*]Partial funding provided by Norway Opening Universities (NOU, *http://norgesuniversitet.no/*) through project P7703.

[†]Department of Informatics, University of Bergen, P.O.Box 7800, N-5020 Bergen, Norway. Email: `{khalid.mughal,yngvee}@ii.uib.no`

[‡]Nansen Environmental and Remote Sensing Center, Thormøhlens gt. 47, N-5006 Bergen, Norway. Email: `torill.hamre@nersc.no`

of presentations, as long as the content conforms to the structural requirements of the pattern.

In this report we present a system that can generate multiple online courses from the same presentation pattern when supplied with just the content for the new courses. For exposition purposes, we will use a simplified version of the presentation pattern called the Course Pattern that is used for online Java programming courses offered by the Department of Informatics, University of Bergen. Until recently, the maintenance of our web pages for an online course and its content required programming background on the part of the course designer. In order to simplify setting up new online courses, we needed solutions that do not require skills beyond procurement of the content or data for the course. The content is specified in XML and its structure is dictated by the Course Pattern. The course designer need only supply the content in order to create an online course. The system takes care of the rest: dynamically generating the web pages for the course and making them accessible to the users.

2 Generating Web-based Presentations

The work flow model of the Dynamic Presentation Generator (DPG) system is shown in Fig. 1. It comprises two phases: a static phase and a dynamic phase. In the static phase, the Repository Administration Tool (RAT) validates the data in XML files against the content structure specified in the presentation pattern. Only validated data is stored in the native XML database Apache Xindice [1]. RAT is also used to retrieve the data from the database for updating purposes.

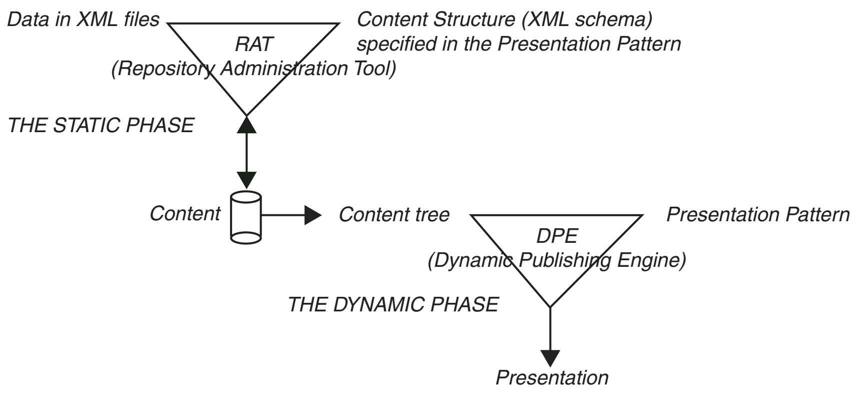

Data in XML files
RAT
(Repository Administration Tool)
Content Structure (XML schema) specified in the Presentation Pattern
THE STATIC PHASE
Content
Content tree
DPE
(Dynamic Publishing Engine)
Presentation Pattern
THE DYNAMIC PHASE
Presentation

Fig. 1. The Workflow Model of the DPG System

The core of the dynamic phase is the Dynamic Publishing Engine (DPE). Given the content tree and the corresponding presentation pattern, the DPE renders the web pages that comprise the presentation. The DPE dynamically generates a web page in response to a browser request. The content tree is created from the content in the XML database at the start of the web application. Data for a browser request is retrieved from the content tree. Formatting of a web page is done according to the presentation pattern specification.

An obvious advantage of this workflow model is that different content and presentation patterns can be mixed and matched to create different presentations, as long as the content conforms to the presentation pattern.

In the next section we describe how the various aspects of a presentation are specified by means of a presentation pattern.

3 Specifying Presentations

The different aspects of a presentation are primarily defined by two specifications: the *Presentation* and the *Presentation Pattern* specifications. Together they capture the salient features of a presentation that the DPE can generate dynamically. For the Course Pattern, its Presentation specification and Presentation Pattern specification are shown in Fig. 2 and Fig. 3, respectively. The rest of this section will describe these specifications.

```
<presentation>
 <title>INF100-F Introduction to programming</title>
 <presentation-pattern>
  coursePattern
 </presentation-pattern>
 <stylesheet>default.css</stylesheet>
 <content>
  <resource path="/inf100f" name="startpage" />
  <resource path="/inf100f" name="assignments" />
  <resource path="/inf100f" name="resources" />
  <resource path="/inf100f" name="archive" />
  <resource path="/inf100f" name="contact" />
 </content>
 <layout-mapping>
  <perspective-name>default</perspective-name>
  <layout-template>course.jsp</layout-template>
 </layout-mapping>
 <logo>uib_logo2.jpg</logo>
 <header-text>
  INF100-F Introduction to programming
 </header-text>
 <footer-text>University of Bergen 2004</footer-text>
</presentation>
```

Fig. 2. The Presentation Specification for the Course Pattern

3.1 The Presentation Specification

The Presentation specification specifies the following aspects, exemplified below from Fig. 2:

- The title of the presentation (`INF100-F Introduction to programming`)
- The presentation pattern to use for this presentation (`coursePattern`) (*see section 3.2*)
- The Cascading Stylesheet to format the content (`default.css`)
- The content for the presentation (specified by the `<content>` element)
- The layout for the presentation, specified by mapping the perspectives to their layout templates (the `default` perspective is mapped to the `course.jsp` page) (*see section 3.2*)
- Other global layout features like a logo, a header, a footer, etc. that should be included in the presentation

Note that the Presentation specification can be customized with regard to what presentation pattern should be used, what formatting properties to use for the web page, which content to associate with the presentation, and which layout to use for the presentation components in a browser window. This parameterization of the different aspects of a presentation allows great flexibility in creating presentations.

3.2 The Presentation Pattern Specification

The Presentation Pattern specification specifies the following aspects, exemplified below from Fig. 3:

- The XML schema that defines the content structure. In Fig. 3 the content structure is defined by the XML schema in the `courseContentSchema` resource. See section 3.2.1 for details on the content structure used with the Course Pattern.
- Presentation units called *views* that map to specific elements of the content structure. In Fig. 3 two views (`sections` and `subsections`) are specified. See section 3.2.2 for details on how views are mapped to the content structure for the Course Pattern.
- Presentation units called *content viewers* that can display the content structure denoted by views. A view must first be associated with a content viewer. In Fig. 3 one content viewer (`mainContentViewer`) is specified. It is associated with the `sections` and `subsections` views of the Course Pattern. See section 3.2.3 for details on the relationship between the content viewer and the views in the Course Pattern.
- Presentation units called *perspectives* that are containers for views and content viewers. A perspective allows views and content viewers to be grouped together. It also specifies the *navigational dependencies* between the views. In Fig. 3 one perspective (named `default`) is specified, containing the `sections` and `subsections` views and also the `mainContentViewer`. See section 3.2.4 for details on the perspective specified for the Course Pattern.

```
<presentation-pattern name="coursePattern">
 <content-structure>courseContentSchema</content-structure>
 <views>
  <view name="sections">
   <content-path>section</content-path>
   <identified-by>@title</identified-by>
  </view>
  <view name="subsections">
   <content-path>subsection</content-path>
   <identified-by>@title</identified-by>
  </view>
 </views>
 <content-viewers>
  <content-viewer name="mainContentViewer">
    <view-mapping>
     <view-name>sections</view-name>
     <content-path>para</content-path>
     <transform-by>
      courseContentTransformation
     </transform-by>
    </view-mapping>
    <view-mapping>
     <view-name>subsections</view-name>
     <content-path>*</content-path>
     <transform-by>
      courseContentTransformation
     </transform-by>
    </view-mapping>
  </content-viewer>
 </content-viewers>
 <perspectives>
  <perspective name="default">
   <view-name>sections</view-name>
   <view-name>subsections</view-name>
   <content-viewer-name>
    mainContentViewer
   </content-viewer-name>
   <view-dependency>
    <view-name>subsections</view-name>
    <depend-on>sections</depend-on>
   </view-dependency>
  </perspective>
 </perspectives>
</presentation-pattern>
```

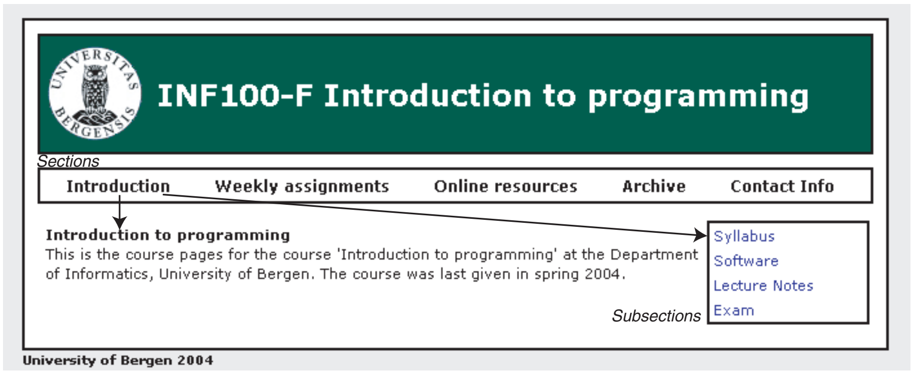
Fig. 3. The Presentation Pattern Specification for the Course Pattern

3.2.1 The Content Structure

The Presentation Pattern specification for the Course Pattern in Fig. 3 specifies the XML schema which defines the content structure. Each content resource specified in the Presentation specification in Fig. 2 must be valid according to this schema. For the Course Pattern, the schema requires the course structure to be defined by XML elements `<section>`

and `<subsection>`. How the resource named `startpage` in the `<content>` element in Fig. 2 is defined as a section called `Introduction`, together with its subsections, is shown below:

```
<section title="Introduction">
 ...
 <subsection title="Syllabus"> ... </subsection>
 <subsection title="Software"> ... </subsection>
 <subsection title="Lecture Notes"> ... </subsection>
 <subsection title="Exam"> ... </subsection>
</section>
```



Introduction to programming
This is the course pages for the course 'Introduction to programming' at the Department of Informatics, University of Bergen. The course was last given in spring 2004.

Fig. 4. Content Structure

Fig. 4 shows how the section specified above and its subsections are displayed in the web page. As can be seen from Fig. 4, the presentation has five sections that are defined by the resources specified in Fig. 2. Note that the titles of sections and subsections designate hyperlinks.

3.2.2 Views

Navigation aspects of a presentation are specified using views. A view contains a set of *navigation elements* that enable navigation between specific parts of the content. In our Course Pattern, readers should be able to navigate the sections (via the `sections` view) and also the subsections within each section (via the `subsections` view). In Fig. 3, the `sections` view is mapped to a `section` element in the content structure as shown in the following snippet:

```
<view name="sections">
 <content-path>section</content-path>
 <identified-by>@title</identified-by>
</view>
```

The `sections` view is a list of navigational elements, where each such navigational element is a hyperlink to a section with the section title as the hyperlink text (specified in the `<identified-by>` element). Analogously, the `subsections` view is a list of hyperlinks representing the subsections in a particular section. In Fig. 5, the two views of the Course Pattern are clearly identified. The `subsections` view is showing the subsections under the section entitled "Weekly assignments".

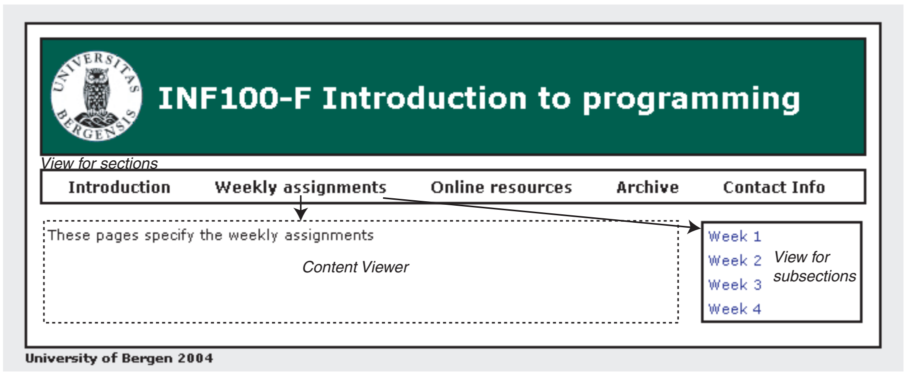


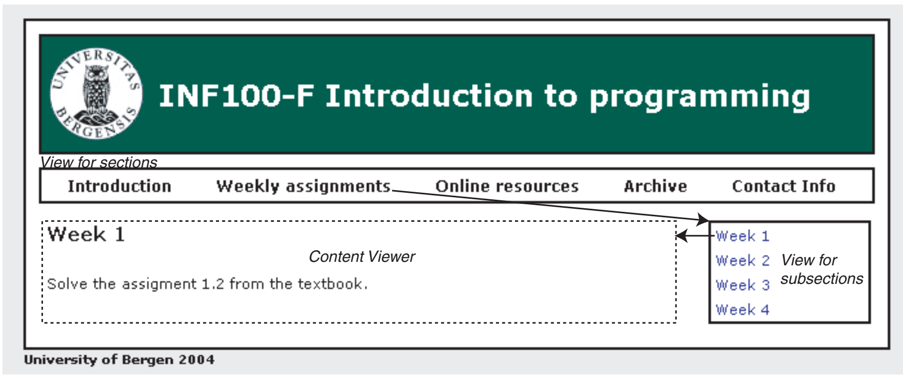
Fig. 5. Views and Content Viewer

3.2.3 Content Viewers

Views provide the mechanism to navigate the presentation units, but a view does not specify how its underlying content should be displayed. This is handled by a content viewer. A content viewer is associated with a view and this association signifies that the content structure denoted by the view is rendered by the content viewer. If the content of the view must be transformed before being presented, a *transformation document* must also be specified. In Fig. 3, the `mainContentViewer` is associated with the `sections` view:

```
<view-mapping>
 <view-name>sections</view-name>
 <content-path>para</content-path>
 <transform-by>
  courseContentTransformation
 </transform-by>
</view-mapping>
```

In the specification above, the `mainContentViewer` will display the `<para>` element of the section that is associated with the `sections` view. However, the content will be transformed according to the transformation document `courseContentTransformation` before it is displayed. In Fig. 5, selecting the "Weekly assignment" section displays the `para` element of this section. However, if the subsection "Week 1" is now selected in Fig. 5, the display changes as shown in Fig. 6. The content viewer displays the contents associated with the last hyperlink that was selected from one of its views.



Week 1

Solve the assigment 1.2 from the textbook.

Fig. 6. Navigation

3.2.4 Perspectives

Views and content viewers can be nested in containers. Such a container then constitute a perspective of the presentation pattern. An important aspect of a perspective is the ability to define *dependencies* between views. The `subsections` view depends on the `sections` view in the Course Pattern. This dependency forces the `subsections` view (the dependee view) to list subsections within a specific section in the `sections` view (the depender view). The subsections listed by the `subsections` view will be from the section which has been selected in the `sections` view. The implications of this dependency are illustrated by the screen shots in Fig. 4 and Fig. 5. Selecting a particular section updates the `subsections` view with the subsections from the selected section.

Each perspective is mapped to a *layout template* (a JSP page specified in the Presentation specification, see Fig. 2) that renders the visual representation of views and content viewers contained in the perspective. The layout template also defines the placement of the rendered views and content viewers in the perspective. The screen shots in this section demonstrate the results of applying the layout template for the Course Pattern.

4 Current Status and Future Work

The first version of the DPG system was used to create online Java programming courses in spring 2004 and spring 2005 at the Department of Informatics, University of Bergen [5]. Details on the implementation of this version can be found in [7]. Both the DPG system and the presentation specifications have now gone through a number of iterations. They have provided the proof-of-concept for presentation patterns, as well as hands-on experience from running and maintaining the online courses.

The current system is implemented in Java using Eclipse [6], and runs under the Apache Tomcat web server [1]. Both Java and Tomcat are readily available for installation on nearly all platforms. A course administrator will need to deploy Tomcat, but does not need Eclipse. A student will only need a web browser to access an online course delivered by the DPE.

The RAT facility has a web-based GUI that allows uploading of initial content from XML files and stores the content as a set of resources in the Xindice database. The tool also incorporates a general-purpose editor for content inspection and modification. For convenience, the database files and any associated resources (for example, images) are stored as part of the DPE web application. If any of the content is modified by the administration tool, the publishing engine automatically updates the presentation.

One high-priority task is to create new presentation patterns. Typical examples of new patterns would be for slide shows, for interactive presentation of a lecture or for "webifying" articles and books. The main challenge will be achieving this goal through reuse of web-based presentation components. The aim is not just to create dynamic HTML pages, but also to develop functionality that exploits the content in new ways. The administration tool will also be extended to streamline the process for online course development, for example, by providing better support for content generation and customization based on presentation patterns.

Our experience has shown that there are several advantages to using presentation patterns to create online courses. For instance, an initial investment in defining a suitable navigation structure and visually appealing layout can be capitalized on in later courses, as these aspects of a presentation are captured in the presentation pattern. From a course administration point of view, no programming experience is needed to prepare and update the content, and web-based tools are available for content generation and maintenance. In terms of cost and effort, the threshold to deploy this system is low compared to other such systems.

5 Related Work

A number of methodologies, technologies and tools exist for organizing content for web-based presentations. Design and development of data-intensive web applications is an area of active research (WebML [3], Strudel [9], Areaneus [4]). These and other research teams are attacking the problem of defining the structure, navigation and layout of a web presentation using various data modelling formalisms. For instance, the WebML team has developed a platform- and product-independent methodology, process and notation for the development of data-intensive web applications. In WebML, data (i.e. content) structure is defined using the E-R model, while the WebML notation is used to specify how specific (selected) parts of the content are mapped to hypertext elements such as units, links and pages. WebML also defines how these elements are organized into composite elements such as nested pages and site views. In our approach we use XML both for specifying the content structure and for organizing and mapping the content to views and content viewers.

The underlying data structure in applications modelled by WebML differs from that of the content used in the DPG system. WebML applications reported in the literature, e.g. [3] and [4], are driven by relational databases where table fields are typically limited-length strings or numbers. This is not to say that these data structures are not complex, just that their elements are small and more uniform compared to the content, for example, in e-learning. The units in e-learning tend to be larger and more diverse, being composed by nesting simple and compound elements into large hierarchical data structures. In contrast, the content used by the DPG system is stored in a native XML database and its structure is only limited by the page rendering transformation that is implemented for the presentation pattern and applied by the DPE at runtime.

6 Conclusion

The DPG system presented in this report promotes reuse of content (i.e. data) *and* presentation patterns for developing online courses. Non-programmers can readily use the system to create online courses once the presentation pattern has been defined. In many online offerings, where a full-blown e-learning system would be an overkill, our system can be used with advantage. The system is also developed using non-propriety software, minimizing the cost of its deployment.

We expect significant savings in development and maintenance of online courses by using our system. The aim is to develop and use new presentation patterns in the creation of different types of presentations.

References

[1] Apache Tomcat, *http://jakarta.apache.org/tomcat/*.

[2] Apache Xindice, *http://xml.apache.org/xindice/*.

[3] S. CERI, P. FRATERNALI, A. BONGIO, M. BRAMBILLA, S. COMAI AND M. MATERA. *Designing Data-intensive Web Applications*, Morgan Kaufmann, San Francisco, 2003.

[4] S. CERI, P. FRATERNALI, M. MATERA AND A. MAURINO. Designing Multi-Role, Collaborative Web Sites with WebML: a Conference Management System Case Study. *Proceedings of First International Workshop on Web Oriented Software Technology (IWWOST'01)*, pp. 130 - 152, 2001.

[5] K. CRUICKSHANKS. *Verktøy for generering av XML-baserte presentasjonar: JPGen - Java Presentasjons generator.* Master thesis, Department of Informatics, University of Bergen, 2003 (In Norwegian).

[6] Eclipse, *http://www.eclipse.org/*.

[7] Y. ESPELID. *Dynamic Presentation Generator.* Master thesis, Department of Informatics, University of Bergen, 2004.

[8] M. F. FERNANDEZ, D. FLORESCU, A. Y. LEVY AND D. SUCIU. Catching the Boat with Strudel: Experiences with a Web-Site Management System. *Proceedings of SIGMOD Conference*, pp. 227-263, 1998.

[9] P. MERIALDO, P. ATZENI AND G. MECCA. Design and Development of Data-Intensive Web Sites: The Araneus Approach. *ACM Transactions on Internet Technology*, Vol. 3, No. 1, pp. 49-92, 2003.