HPG: A Tool for Presentation Generation in WIS

Bas Rutten, Peter Barna, Flavius Frasincar, Geert-Jan Houben, and Richard Vdovjak

Eindhoven University of Technology, Department of Computer Science

PO Box 513, 5600 MB, Eindhoven, the Netherlands

{brutten,pbarna,flaviusf,houben,richardv}@win.tue.nl

ABSTRACT
Web Information Systems (WIS) support the process of retrieving information from sources on the Web and of presenting them as a hypermedia presentation. Most WIS design methodologies focus on the engineering of the abstract navigation (hyperlinks). The actual presentation generation is less supported. Hera is one of the few WIS methodologies that offer a tool for presentation generation (HPG). The HPG transforms RDF data obtained as the result of a query into a Web presentation suited to the user (in HTML or WML).

Categories and Subject Descriptors
H.1 **[Information Systems]**: Models and principles; H.5.4 **[Information Interfaces and Presentation]**: Hypertext/Hypermedia Architectures, Navigation; D.2.2 **[Software Engineering]**: Design tools and techniques

General Terms
Design, Management

Keywords
WIS, Hypermedia, RDF(S), XSLT, Presentation generation

1. INTRODUCTION
The data-intensive and dynamic structure of modern WIS call for a rigorous development process, that takes into account the fact that information from various heterogeneous sources needs to be delivered to a variety of users using a variety of platforms and network connections. Several frameworks for this already exist, e.g. WebML [2] and Hera [3]. This paper demonstrates the software application called the Hera Presentation Generator (HPG) and its role in the (semi-automatic) generation of hypermedia presentations according to this framework.

2. HERA METHODOLOGY
The Hera methodology distinguishes two phases in the generation of a WIS:
1. During the integration and data retrieval phase, data is collected from different sources and converted to a uniform representation (beyond the scope of this paper).
2. During the presentation generation phase, the retrieved data is transformed into a hypermedia presentation in a format suited to a user. This phase has two modeling aspects:
    a. The Application Layer defines the abstract hypermedia (navigation) view over the data in terms of an Application Model (AM).
    b. The Presentation Layer defines the presentation layout that together with the definitions from the Application Layer is needed for the generation of a presentation for a concrete platform.

3. PRESENTATION GENERATION
The presentation generation phase of the Hera methodology has been implemented as a Java application with connections to Microsoft Visio, called the Hera Presentation Generator (HPG). This section gives an overview of this application.

3.1 CM and AM construction
The first step of creating a presentation is the definition of a Conceptual Model (CM, a definition of the data domain) and an Application Model (AM). The CM consists of *concepts* and *concept properties* that can be used to link concepts to media items, or to define relations between concepts. Figure 1 shows an example CM, based on the database of the Topia project[1], containing data about art from the Rijksmuseum[2], created with HPG's CM Builder using a specially designed Microsoft Visio template. Concepts are represented by ovals, the media items by rectangles, and the *concept properties* by arrows.
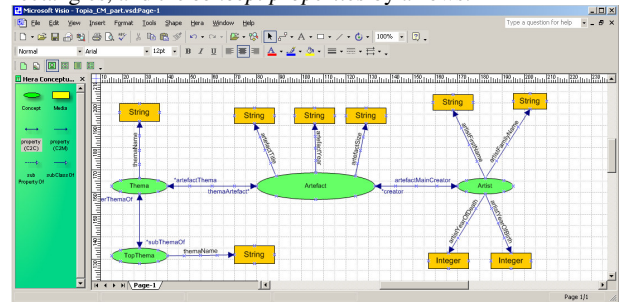

Figure 1: Example Conceptual Model

The designer can create an AM based on the CM to define the structure of a presentation. For this, the AM Builder has been developed on the basis of another Visio template. An AM defines the navigational structure of the presentation (at schema level). It consists of *slices* (a slice is a meaningful representation of some media items) and *slice properties* that together define the navigation vocabulary. There are two types of slice properties: slice composition (a slice encloses another slice) and slice navigation (a slice is the anchor of a hyperlink to another slice). To support user adaptation, slices can have *appearance conditions*. These conditions enable two kinds of adaptation: *link hiding* and *conditional inclusion of fragments* [1] (slices, in this context). Both the AMs and the CMs created using the Builders can be exported to RDFS format, for use with the HPG core application. Figure 2 shows an example AM, with two top-level slices (the large rounded rectangles with the circle segments) that each have a number of nested slices (the small ovals), and examples of *slice navigation* (represented by arrows).
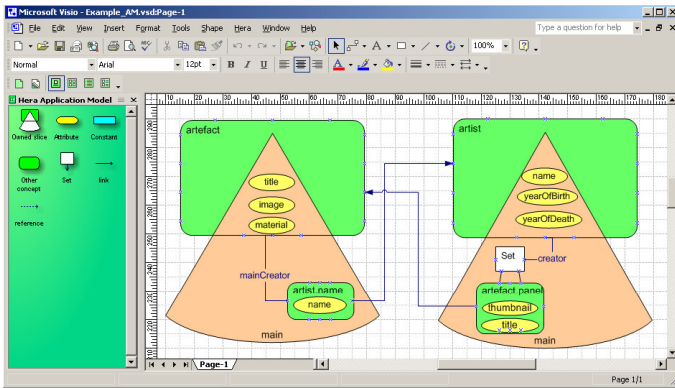
---

[1] http://www.telin.nl/CE/Topia/index.htm

[2] http://www.rijksmuseum.nl

Figure 2: Example Application Model
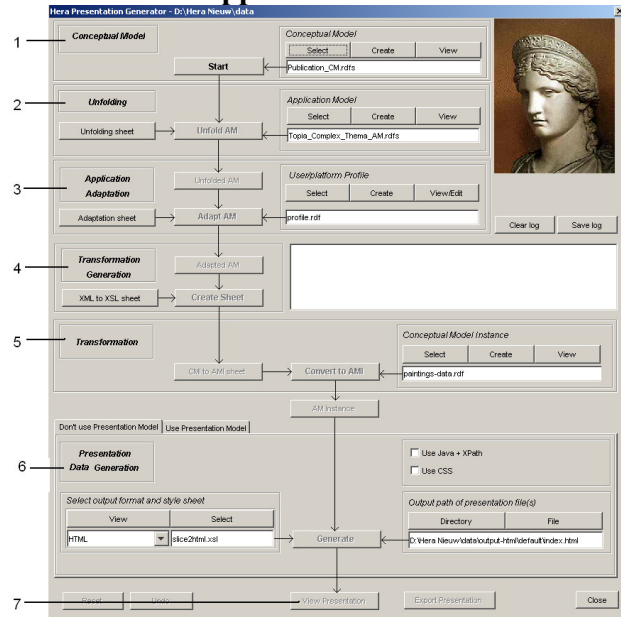
3.2 HPG Core Application

Figure 3: HPG "advanced" user interface screen shot

The core of HPG is a Java-based application that allows the designer to quickly construct and customize a hypermedia presentation based on an AM and a set of RDF input data (Conceptual Model Instance, CMI), by carrying out a number of transformation steps. Each step takes one or more RDF(S) files as input and uses an XSLT stylesheet to construct an RDF file, or in case of the last step, a presentation suited to the user's platform (HTML or WML). For carrying out the XSLT transformations, we use the Saxon interpreter[3]. Figure 3 shows a screenshot of HPG's methodology demonstrator where one can view every input, output, and intermediary file used by the system, as well as the XSLT stylesheets. In addition to this "advanced" interface, HPG provides a wizard interface for regular users. The different steps of the generation process (corresponding to the numbers in Figure 3) are as follows:

1. The user starts by selecting a CM or creating a new one. The lists of CMIs and AMs will be filtered so that they only contain the files based on the selected CM.

2. The user then selects an existing AM or creates a new one (based on the selected CM) using HPG's graphical AM Builder.
3. In *Application Model Adaptation* the first part of adaptation is carried out by adapting the AM to a user/platform profile. The appearance conditions in the AM are compared to the values in this profile, and slices and links that do not have their conditions fulfilled are removed from the presentation.
4. In *Application Model Instance Transformation Generation* an XSLT stylesheet is created that is used by the next step.
5. In *Application Model Instance Generation* the stylesheet from the previous step is used to instantiate (fill) the AM with data from a CMI (input data) chosen by the user. The result of this step is the Application Model Instance (AMI).
6. In *Presentation Data Generation* the AMI is converted into a presentation in the format selected by the user (currently HTML and WML are supported). Optionally, a Presentation Model (PM) can be used to adapt the layout of the presentation, e.g. to user preferences.
7. The presentation generation process is now complete, and the user can view the presentation in HTML or WML.
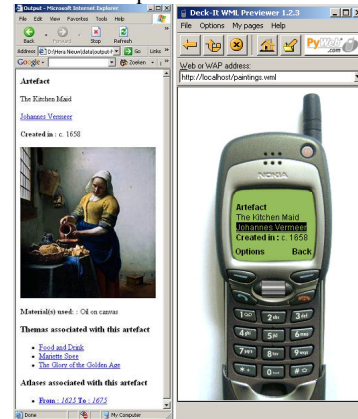

Figure 4: Screenshots of generated presentations

4. CONCLUSIONS AND FUTURE WORK

HPG, the Hera Presentation Generator, allows users to quickly generate hypermedia presentations, and to customize them by adaptation to user/platform profiles, and by supporting various output formats. In the future, we plan to create presentations dynamically (i.e. we only create a part when requested by the user instead of creating the entire presentation beforehand). This will facilitate the inclusion of *adaptivity* (changing the presentation based on the user's actions) and the support for more advanced (compared to hyperlink navigation) user interaction, e.g. allowing the user to submit forms. The new dynamic version of HPG will be implemented in Java, using the Jena and Sesame APIs.

5. REFERENCES

[1] Brusilovsky, P. (2001). *Adaptive Hypermedia*, User Modeling and User-Adapted Interaction, Vol. 11, No. 1-2, pp. 87-110.

[2] Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., and Matera, M. (2003). *Designing data-intensive Web applications*, Morgan Kaufman.

[3] Vdovjak, R., Frasincar, F., Houben, G.J., and Barna, P. (2003). *Engineering Semantic Web Information Systems in Hera*, JWE, Vol. 2, No. 1&2, pp. 3-26, Rinton Press.

---

[3] http://saxon.sourceforge.net