RDF/XML-based Automatic Generation of Adaptable Hypermedia Presentations

Flavius Frasincar, Geert-Jan Houben, Peter Barna, and Cristian Pau
Eindhoven University of Technology
PO Box 513, NL-5600 MB Eindhoven, the Netherlands
{flaviusf, houben, pbarna, cpau}@win.tue.nl

Abstract

Hera is a design methodology that supports the development of web information systems. It is a model-driven method that distinguishes four steps: data retrieval, application model generation, application model instance generation, and presentation data generation. Data retrieval populates the conceptual model with data. In the application model generation the navigational aspects of the application are specified in the application model. Also, the application model needs to be adapted for different user/channel profiles. In the third step of the Hera method, the application model is populated with the retrieved data. The last step considers the physical aspects of the presentation: the retrieved data wrapped in application logic is translated to different implementation platforms (e.g. HTML, WML, SMIL). Having in mind the advantage of web application interoperability we chose to implement an experimental prototype for the Hera method using RDF(S), the foundation of the Semantic Web.

1. Introduction and related work

The Web is the most popular source of information today. As a result, a key success factor for modern information systems is their web presence. From user's point of view the one-size-fits-all paradigm is not acceptable anymore. The content needs to be personalized for different users based on their preferences. Moreover such information systems should enable multichannel access as more and more users will reach them using different platforms (e.g. PC, PDA, WAP phone, WebTV) and different network speeds (e.g. dial-up modem, network copper cable, network fiber optic cable). Due to these requirements it has become increasingly important to have an engineered design of web information systems.

Research literature provides good references like Relationship Management Methodology (RMM) [9] and Object-Oriented Hypermedia Design Methodology (OOHDM) [15] for hypermedia design methodologies that offer guidelines to the developer of a Web Information System. However, RMM doesn't consider the multichannel/multiuser aspects of the present Web and OOHDM offers weak support for automation. Recent research studies like WebML [4], XAHM [3], UWE [13], XWMF [11], and Hera [8] give design steps towards a (partially) automated generation of hypermedia presentations (from retrieved data) using modern web technologies (e.g. XML, RDF) or object-oriented modeling languages (e.g. UML). All these methodologies are model-based, the role of metadata being crucial for automation support. Adaptation with respect to the user's preferences or device capabilities is considered integrant part of the design process.

With more than 3 billion pages the Web is the most important source of information. The emerging Semantic Web (SW) [1] aims at providing a unique (semantical) view over this data that will ensure web application interoperability. The foundation of the SW is the Resource Description Framework (RDF) [14] and the RDF Schema (RDFS) [2]. RDFS extends RDF by providing means to describe domain vocabularies. XWMF consists of an extensible set of RDF schemata and descriptions to model web applications. The choice to represent models in RDF(S) is also done in Hera. In contrast to XWMF which is only a modeling framework, Hera provides both a modeling framework and a methodology for developing web applications.

2. Method and tools

A primary focus of the Hera project [8] is to support Web Information System (WIS) design and implementation. In response to a user query, a WIS should automatically generate a hypermedia presentation for data possibly coming from heterogeneous sources. The generated hypermedia presentations need to be tailored (adapted) for different device (network, display) capabilities and different user preferences.

Lacking a mature web ontology language, we chose to represent Hera models/instances in RDF(S) by providing appropriate extensions. In contrast to XML and UML, RDF(S) was designed by W3C as the Web metadata language being better able to deal with the semistructure nature of Web metadata. RDF(S) is a flexible (supporting schema refinement and description enrichment) and extensible (allowing the definition of new resources/properties) framework that enables web application interoperability. An example of application interoperability is the usage of different navigation ontologies for a given application domain ontology. In Hera, model instances are represented in plain RDF that will be validated against their associated models (schemas) represented in RDFS. Using RDF(S) as the underlying representation formalism enables us to reuse existing RDF(S) vocabularies like the User Agent Profile (UAProf) [16], a Composite Capability/Preference Profiles (CC/PP) [12] vocabulary for modeling device capabilities and user preferences [7]. The syntax carrier is RDF/XML, the XML serialization of RDF. While RDF(S) seems to foster the specification of Hera's different models, there is, to our knowledge, no full-fledged RDF(-aware) transformation processor. Nevertheless, the Hera models and their instances can be treated as plain XML representations on which an XSLT processor can perform different transformations based on stylesheets. For our purpose this approach proved to be satisfactory as we didn't use RDF(S) inference rules (e.g. transitivity of inheritance) in the transformation specification.

Based on a small set of data made available by the Web site of the Rijksmuseum in Amsterdam, an experimental prototype for the Hera method was developed. At the beginning of the implementation we used Xalan 1.2D02, an XLST processor that supports XPath 1.0 and XSLT 1.0. As Xalan didn't fulfill the high demands of our application (e.g. multiple outputs for one stylesheet, need of procedural constructs like variable assignment and loop operators) we replaced it in the last step of the Hera method (presentation data generation) with Saxon 7.0, an implementation of the more powerful XPath 2.0 and XSLT 2.0 [10].

Figure 1 gives an overview of the Hera method. The four steps of the Hera method appear as numbers (labels) on the continuous arrows. Step 2 and step 3 have both two substeps marked by the second digit notation. Substep 2.2 has two inputs, the application model (unfolded) and the user/platform profile, while step 4 has three alternative outputs for different code generators (HTML, WML, SMIL). There are two types of dashed arrows: "is used by" to express that an RDFS model is used by another RDFS model and "has instance" to denote that an RDFS model has as instance a specified RDF model. The figure has two orthogonal dimensions: generic/specific and static/dynamic. The first dimension generic/specific differentiates between the generic, application (domain) independent models/transformations, and the specific, application (domain) dependent models/transformations. The second dimension static/dynamic underlines the static, fixed, representations versus dynamic representations, i.e. newly generated representations for different retrieved data sets (of a given application domain). Note that in the middle of Figure 1 there is only dynamic (application-)specific information that will change with each retrieved data set.

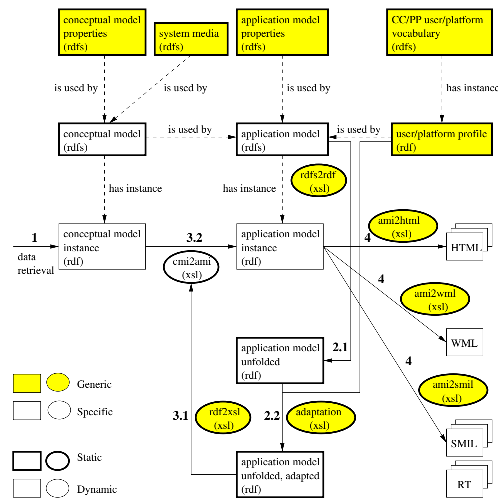

Figure 1. Method

2.1. Data retrieval

In the data retrieval step the conceptual model is populated with the retrieved data. In case that data is coming from heterogeneous sources it is the task of a mediator system (outside the scope of this paper) to integrate the data.

Conceptual model

The conceptual model (CM) provides a uniform schema over the data sources. CM describes the domain ontology of the web application. The basic elements of the CM are concepts and concept properties. We distinguish two types of concept properties: concept attributes which relate concepts to media items and concept relationships which provide associations between concepts.

As shown in Figure 1 the conceptual model is represented in RDFS and it uses two other RDFS descriptions: CM properties and system media types.

CM properties describe the `cardinality` and `inverse` of concept relationships. Knowing in advance if one or more instances are to be retrieved and the ability to traverse concept relationships in both directions are useful features that will be exploited in the following step: application model generation.

The `Media` class is the root of the multimedia ontology. It is further refined using the RDFS subclass mechanism in `Text` and `Image`. `Text` has two subclasses `Integer` and `String`. Each media class has its own properties, `Text` has `length` (expressed in number of characters) and `Image` has `width` and `height` (expressed in pixels). We restricted ourselves to these types for reasons of simplicity, other media types like audio and video could be added seamlessly to this multimedia ontology.

Conceptual model instance generation

The conceptual model is the interface between the data retrieval and the presentation generation. A data retrieving system will associate type information to the retrieved data instances using concepts from the CM. These data instances together with the associated types form the so-called conceptual model instance. The conceptual model instance is an RDF description which validates the conceptual model RDFS schema.

2.2. Application model generation

In the application model generation step the navigational aspects of the application are specified in the application model. This step is composed of two substeps: application model unfolding and (unfolded) application model adaptation.

Application model

The application model (AM) describes the navigation ontology of the web application. The basic elements of the AM are slices and slice properties. Slices are meaningful navigation units that group media items possibly coming from different concepts. We distinguish two types of properties between slices: slice composition, a slice contains another slice, and slice navigation, a slice is an anchor of a hyperlink to another slice. The most primitive slice is the media slice containing only a media item. At the top of the composition hierarchy there are the top level slices, slices that correspond to information to be presented at once on the display.

Figure 1 illustrates that the application model is represented in RDFS and it uses two other RDFS descriptions: AM properties and user/platform profile. AM properties describe the slice hierarchy (based on the RDFS subclass mechanism) and the various slice properties. At the top of the slice hierarchy is the class `Slice`. There are four slice properties: `owner` associates a slice with a concept, `slice-ref` refers to slice composition, `link` defines slice navigation, and `media` points to media items. Based on the `owner` property the AM is built on top of the CM, a feature that will enable the transformation of the CM instance into an AM instance. The `slice-ref` property that connects slices belonging to different concepts (different `owner`) has a `relationship-ref` property to refer to the associated concept relationship. `relationship-ref` can be generalized to a sequence (a path) of concept relationships to connect (for navigation purposes) concepts that are not directly linked in the CM. In order to simplify the AM description the classes/properties that model the notions of set of slices and set of links are omitted from this presentation.

User/platform profile

Adaptation to the AM can be done by adding appearance conditions to slices. These conditions are attached to the `slice-ref` property in order to specify if the referred slice is visible at this particular point or not. In this way the same slice can be visible in one context and invisible in another one. The conditions are using attribute-value pairs stored in the user/platform profile. Composite Capability/Preference Profile (CC/PP) offers a framework to describe vocabularies to model device capabilities/user preferences. In order to build the user/platform vocabulary two CC/PP vocabularies were used: the existing User Agent Profile (UAProf) for modeling device capabilities (e.g. `ImageCapable` attribute) and a vocabulary (e.g. `ExpertiseLevel` attribute) for describing user preferences. As Figure 1 suggests, the user/platform profile is an RDF instance of the considered RDFS user/platform vocabulary.

Application model unfolding

The application model gives the input data for a transformation that populates the application model with retrieved data. In order to ease the description of this transformation stylesheet the RDFS application model needs to be unfolded to its RDF instance representation. This transformation called `rdfs2rdf` in Figure 1 is useful because XSLT is designed for (XML) instance transformations and not for (RDFS) schema transformations. By unfolding the RDFS application model, properties are moved inside the subject classes with a value equal to the corresponding object class. This process is repeated for the object classes and so on, until a full skeleton of the RDF application model instance, the so called unfolded application model, is obtained.

Application model adaptation

The unfolded application model is adapted based on the visibility conditions previously specified in the AM. As Figure 1 suggests, the stylesheet used to specify this transformation is named `adaptation`. The adaptation stylesheet suppresses all slice references for which the condition is made invalid. Links pointing to a suppressed slice are also deleted. This transformation has two inputs: the unfolded application model and the user/platform profile. The XSLT `document()` function enables the reading of multiple inputs. In order to evaluate the adaptation conditions, the XSLT `key()` function is used to retrieve attribute values from the user/platform profile. The conditions check if the attribute value is equal to a predefined condition constant.

2.3. Application model instance generation

In the application model instance generation step the AM is populated with the retrieved data. This step is composed of two substeps: the application model instance transformation generation and the application model instance generation.

Application model instance transformation generation

The application model instance transformation generation step is responsible for building the main transformation stylesheet that will populate the AM with the retrieved data. A similar approach [6] (a stylesheet that generates another stylesheet) was used in the previous version (XML-based) of the Hera prototype. In Figure 1 the transformation stylesheet of this substep is called `rdf2xsl`. The input to this transformation is the AM, unfolded and adapted. One should note that such an AM built on top of the CM has all the information necessary to specify a transformation that will convert CM instances to AM instances. The transformation algorithm has two phases: generate all slice instances for the retrieved data (concept instances) (i) and each time a `slice-ref` is met point to the appropriate slice instance (generated in the previous phase) (ii). The naming convention used to appropriately associate slice instances to each other is the following: each slice instance name is obtained by concatenating the corresponding slice name (e.g. `Slice.painting.main`) with the concept instance identifier (e.g. `Painting_ID1`) that owns this slice instance (e.g. `Slice.painting.main_ID1`).

Application model instance generation

In the application model generation the AM is populated (finally) with the retrieved data. The transformation stylesheet resulted from the previous step is applied to a CM instance to produce an AM instance. In Figure 1 the name of this stylesheet is `cmi2ami`. The input/output models and the stylesheet involved in this transformation are dynamic (application-)specific representations. For a given AM, this stylesheet can be applied to any retrieved data set to produce a valid AM instance.

2.4. Presentation data generation

In the presentation data generation step the retrieved data wrapped in application logic is translated to different implementation platforms. Figure 2 presents how three different serializations (HTML, WML, SMIL) appear in three different browsers: HTML browser (Microsoft Internet Explorer), WML browser (Nokia 7110), and SMIL browser (RealOne Player).

Based on a media-directed translation scheme the different media items are displayed differently in the three browsers: normal font is used for strings and italic font is used for integers. For the WML browser the image is not displayed and in order to view the full text the scroll button needs to be used. A WML back button was implemented to simulate the functionality of the existing back button from the HTML/SMIL browsers.

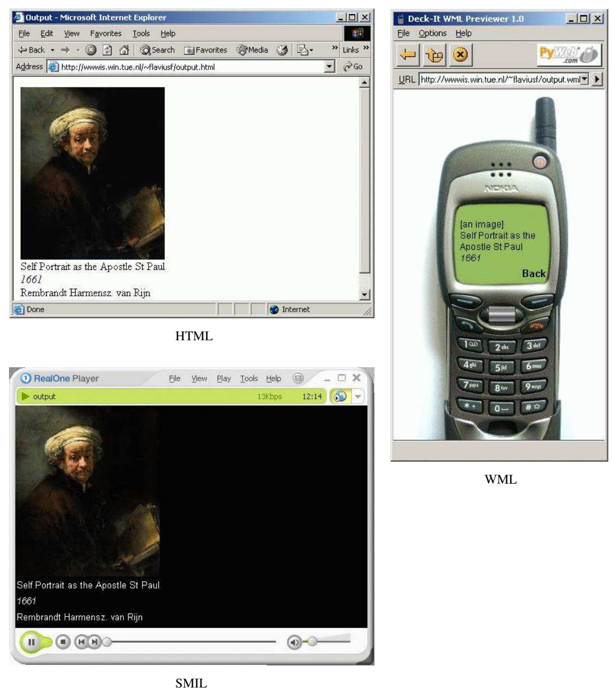

Figure 2. Browsers

Html

In Figure 1 the transformation stylesheet for the HTML serialization is called `ami2html`. Each media item is translated to a paragraph `<p>` containing the actual HTML media item (text or `<img>`). Lists of media items/slices are using in the translation the `<ul>` and `<li>` tags. Hyperlinks are implemented using the `<a>` tag. A HTML presentation is composed from the `index.html` document (starting point of the presentation) and a set of HTML pages each corresponding to a top-level slice. In order to generate multiple outputs the XSLT 2.0 `result-document()` function was used.

Wml

In Figure 1 the transformation stylesheet for the WML serialization is called `ami2wml`. Each media item is translated to a paragraph `<p>` containing the actual WML media item (text or `<img>`). Hyperlinks are implemented using the `<a>` tag. Because lists are not supported in WML, they are implemented as simple sequences of items without any visual cues. To each top level slice corresponds a WML `card`. A WML presentation is composed from a single WML document that contains a set of cards. The first card is the starting point of the presentation.

Smil

In Figure 1 the transformation stylesheet for the SMIL serialization is called `ami2smil`. Each media item is directly translated into its corresponding SMIL media (e.g. `<text>`, `<img>`). Hyperlinks are implemented using the `<a>` tag. Every media item is associated to its corresponding `region`. As a result having lists in SMIL doesn't make sense. A SMIL presentation is composed from a main SMIL document (starting point of the presentation), a set of SMIL documents each corresponding to a top-level slice, and a set of RealText (RT) clips, one per each text media.

3. Conclusions

In this paper we presented Hera, a model-driven methodology for developing WIS and an experimental prototype for the proposed method. Hera is based on two models: the conceptual model and the application model. The Hera methodology identifies three important transformation steps: the adaptation of the application model based on user/platform profile, the population of the adapted application model with retrieved data, and the serialization of the data for different implementation platforms. For representing the different Hera models we used RDF(S), the foundation of the Semantic Web. The extensibility property of RDF(S) enabled a seamless construction of models on top of each other, a useful feature for model-driven transformations. Using the RDF/XML serialization of RDF(S) it was possible to specify the transformation steps in XSLT stylesheets to be used by an XSLT processor.

As a next step we envisage the usage of a web ontology language (like OWL [5]) for the Hera models which can ease not only the model specification but can also provide a higher degree of interoperability between different WIS components.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.

[2] D. Brickley and R. V. Guha. Rdf vocabulary description language 1.0: Rdf schema. W3C Working Draft 30 April 2002.

[3] M. Cannataro, A. Cuzzocrea, C. Mastroianni, R. Ortale, and A. Pugliese. Modeling adaptive hypermedia with an object-oriented approach and xml. In *Second International Workshop on Web Dynamics*, 2002.

[4] S. Ceri, P. Fraternali, and A. Bongio. Web modeling language (webml): a modeling language for designing web sites. *Computer Networks, Ninth International World Wide Web Conference*, 33:137–157, 2000.

[5] M. Dean, D. Connoly, F. van Harmelen, J. Hendler, J. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. Owl web ontology language 1.0 reference. W3C Working Draft 29 July 2002.

[6] F. Frasincar and G.-J. Houben. Xml-based automatic web presentation generation. In *WebNet 2001 World Conference on the WWW and Internet*, pages 372–377. AACE, 2001.

[7] F. Frasincar and G.-J. Houben. Hypermedia presentation adaptation on the semantic web. In *Adaptive Hypermedia and Adaptive Web-Based Systems, Second International Conference, AH 2002*, volume 2347 of *Lecture Notes in Computer Science*, pages 133–142. Springer, 2002.

[8] F. Frasincar, G.-J. Houben, and R. Vdovjak. Specification framework for engineering adaptive web applications. In *The Eleventh International World Wide Web Conference WWW2002 Web Engineering Track*, 2002.

[9] T. Isakowitz, A. Kamis, and M. Koufaris. Extending rmm: Russian dolls and hypertext. In *30th Hawaii International Conference on System Sciences*. IEEE Computer Society, 1998.

[10] M. Kay. Xsl transformations (xslt) version 2.0. W3C Working Draft 16 August 2002.

[11] R. Klapsing and G. Neumann. Applying the resource description framework to web engineering. In *Electronic Commerce and Web Technologies, First International Conference, EC-Web 2000*, volume 1875 of *Lecture Notes in Computer Science*, pages 229–238. Springer, 2000.

[12] G. Klyne, F. Reynolds, C. Woodrow, and O. Hidetaka. Composite capability/preference profiles (cc/pp): Structure and vocabularies. W3C Working Draft 15 March 2001.

[13] N. Koch, A. Kraus, and R. Hennicker. The authoring process of the uml-based web engineering approach. In *First International Workshop on Web-Oriented Software Technology*, 2001.

[14] O. Lassila and R. R. Swick. Resource description framework (rdf) model and syntax specification. W3C Recommendation 22 February 1999.

[15] D. Schwabe, G. Rossi, and S. D. J. Barbosa. Systematic hypermedia application design with oohdm. In *The Seventh ACM Conference on Hypertext, Hypertext'96*, pages 116–128. ACM, 1996.

[16] Wireless Application Protocol Forum, Ltd. Wireless appliciation group: User agent profile. Version 20 October 2001.