Hera: Development of Semantic Web Information Systems

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

Technische Universiteit Eindhoven
PO Box 513, NL-5600 MB Eindhoven, The Netherlands
{houben, pbarna, flaviusf, richardv}@win.tue.nl

Abstract. As a consequence of the success of the Web, methodologies for information system development need to consider systems that use the Web paradigm. These Web Information Systems (WIS) use Web technologies to retrieve information from the Web and to deliver information in a Web presentation to the users. Hera is a model-driven methodology supporting WIS design, focusing on the processes of integration, data retrieval, and presentation generation. Integration and data retrieval gather from Web sources the data that composes the result of a user query. Presentation generation produces the Web or hypermedia presentation format for the query result, such that the presentation and specifically its navigation suits the user's browser. We show how in Hera all these processes lead to data transformations based on RDF(S) models. Proving the value of RDF(S) for WIS design, we pave the way for the development of Semantic Web Information Systems.

1 Introduction

One reason for the popularity of the World Wide Web is that computer applications provide information for a diverse audience on different platforms worldwide and 24 hours a day and people find and view that information with easy-to-use navigation mechanisms. In the typical Web application the author carefully handcrafted a static collection of pages and links between these pages to present to the user information suited for navigation with a Web browser. The success has led to the desire to exploit this user-friendly paradigm in (professional) information systems. Not only has the typical Web application become data-intensive, using for example data generated from databases, but also information system developers want to extend their (traditionally non-Web) applications along the lines of the Web paradigm. This trend has caused the concept of Web application to evolve in the direction of the concept of Web Information System (WIS) [2], a software component that stores and manages information just like a traditional information system, but specifically uses the Web paradigm and its associated technologies. A specific class of Web technologies is that of the Semantic Web (SW) [1] initiative, specially targeting application interoperability. As we show in this paper, one of the SW languages, RDF(S) [3, 12], offers effective support for the design of WIS.

The data in a WIS is retrieved from a heterogeneous and dynamic set of data sources and subsequently delivered as Web or hypermedia presentations to a heterogeneous group of users with different preferences using different platforms to view the presentations. This poses new and different requirements for the design and development of a WIS: from handcrafting to engineering.

In our Hera [6] methodology we single out a number of characteristic aspects of WIS design:

- Most visible in WIS design (and missing in traditional non-Web approaches) is the need to automatically generate Web or hypermedia presentations for the data the WIS delivers. This includes that the system (instructed by its designer) decides on the navigation structure of the presentation, by describing the composition of the presentation objects (e.g. pages) and their connections (e.g. hyperlinks).
- A WIS includes a transparent repository of data obtained from different sources. The designer has to instruct the system how to search and find, retrieve, transform, and combine information. The aspect of integration is often disregarded in WIS design.
- Personalization is a prominent issue in WIS design: delivering the right data in the right way (e.g. device and connection) to the right user. Hera combines presentation generation with user adaptation [4].

Most WIS design methodologies focus primarily on presentation generation. A lot of models that consider adaptation do so only in the context of adaptive hypermedia documents. As notable exceptions among the Web engineering approaches we mention Object Oriented Hypermedia Design Model (OOHDM) [13], Web Modeling Language (WebML) [5], UML-based Web Engineering (UWE) [11], and eXtensible Web Modeling Framework (XWMF) [7]. For its RDF-based nature we consider the last one further. It consists of an extensible set of RDF schemas and descriptions to model Web applications. The core of the framework is the Web Object Composition Model (WOCM), a formal object-oriented language used to define the application's structure and content. WOCM is a directed acyclic graph with complexons as nodes and simplexons as leaves. Complexons define the application's structure while simplexons define the application's content. Simplexons are refined using the subclassing mechanism in different variants corresponding to different implementation platforms.

Hera uses a model-driven approach, specifying the separate aspects of the complete application design in terms of separate models, e.g. for presentation generation, integration, and adaptation. Similar to OMG's Model Driven Architecture, Hera distinguishes between Platform Independent Models (e.g. integration model, presentation model, and application model) and Platform Dependent Models (e.g. presentation model) but in the specific context of hypermedia. In this way if the targeted platform changes the specifications of the application logic remain the same.

The associated Hera framework implies a stepwise approach to get from data retrieval to presentation generation at the level of instances. Those steps constitute a sequence of data transformations that eventually produces the right data in the right format (e.g. HTML, WML, or SMIL). In the demonstrator for the framework we have used RDF(S) to specify the different models and XSLT [10] to transform the RDF data in order to generate the presentations in the right format.

2 Hera Methodology

From the gathering of requirements to the maintenance of the operational application, most information system design methodologies distinguish several phases in the design process. The development of a WIS is different in several aspects, and these aspects are the central focus of the Hera project.

Like other WIS methodologies Hera includes a phase in which the hypermedia (Web) navigation is specified. Hera considers navigation in connection to the data-intensive nature of the modern WIS in its *presentation generation* phase. Before, Hera's *integration and data retrieval* phase considers how to select and obtain the data from the storage part of the application. This includes transforming the data from these sources into the format (syntax and semantics) used in the application. Also, the handling of the interaction from users is specified: querying, navigation, or application-specific user interaction.

2.1 Model-Driven Transformations

Hera's target is to facilitate the automatic execution of the design: it should be possible to program the WIS in such a way that it can automatically execute the process specified by the design. Figure 1 shows how Hera typically views a WIS architecture:

- The *Semantic Layer* specifies the data content of the WIS in terms of a conceptual model. It also defines the integration process that gathers the data from different sources.
- The *Application Layer* specifies the abstract hypermedia view on the data in terms of an application model representing the navigation structure provided in the hypermedia presentation. It also defines the user adaptation in the hypermedia generation process.
- The *Presentation Layer* specifies the presentation details that (with the definitions from the Application Layer) are needed for producing a presentation for a concrete platform, like HTML, WML, or SMIL.

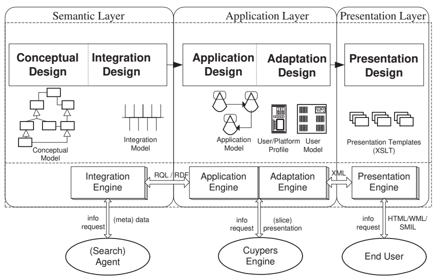

Fig. 1. WIS architecture in Hera

Hera uses several models to capture the different design aspects. Providing clear relationships between the different models, e.g. by expressing one model in terms of an other one, gives a major advantage: model-driven transformations. Populating the different models with data and then transforming them according to the relationships between the models leads to an automatic execution of the design at instance level, and ultimately to the production of the hypermedia presentation.

The models in the phase of integration and data retrieval obtain the data from the sources. In reaction to a *user query* a *conceptual model instance* is produced with the data for which the application is going to generate a presentation: see Figure 2.

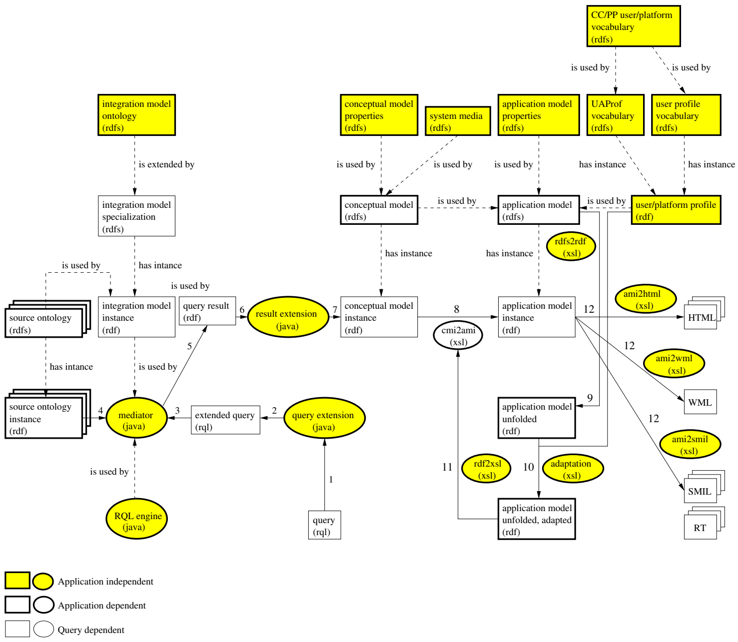

Fig. 2. Hera methodology

The models in the phase of presentation generation generate a hypermedia presentation for the retrieved data: see Figure 2. The result of the user query, represented by the *conceptual model instance*, is transformed into a presentation in the specific format of the user's browser (e.g. HTML, WML, or SMIL).

2.2 Software Demonstrator

The Hera demonstrator uses the example of the design of a virtual art gallery that allows visitors to create on-the-fly exhibitions featuring their favorite painters, paintings, and painting techniques. These hypermedia presentations are produced from the exhibits of different online museums and annotated with descriptions from an online art encyclopedia.

Our demonstrator shows that we have chosen in our methodology to use RDF(S) as the data format for the different Hera models. Hera's model instances are represented in plain RDF with the associated models represented in RDFS. RDF(S) is suitable, since it is a flexible (schema refinement and description enrichment) and extensible (definition of new resources and properties) framework that enables Web application interoperability. We reuse existing RDFS vocabularies like the User Agent Profile (UAProf) [14], a Composite Capability/Preference Profiles (CC/PP) [8] vocabulary for modeling device capabilities and user preferences. A disadvantage is the lack of proper RDF(-aware) transformation processors. Treating the models and instances as plain XML, the XSLT processor Saxon performs the different transformations based on stylesheets. For data retrieval we use RQL [9] and its java-based interpreter Sesame.

3 Integration and Data Retrieval

In the integration phase the designer defines and creates channels connecting the different data sources to the conceptual model. In the data retrieval phase the channels are used to obtain the data that correspond to the conceptual model.

3.1 Conceptual Model

The role of the conceptual model (CM) is to provide a uniform semantic view over the input sources. A CM (Figure 3 - top) uses concepts and concept properties expressed in RDFS to define the domain ontology.

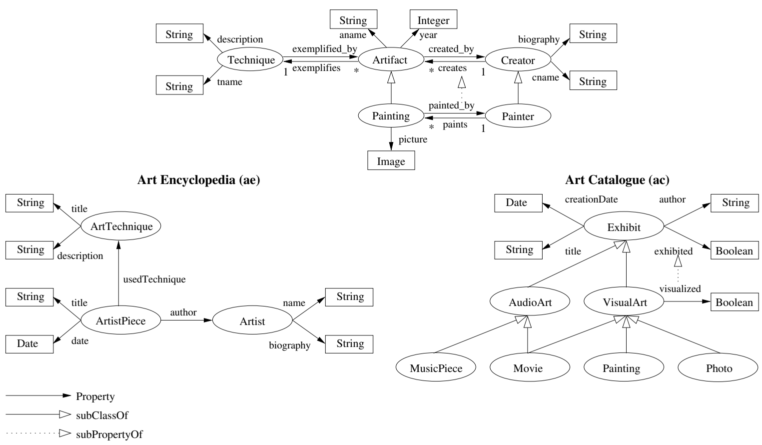

Fig. 3. Conceptual model (top) and integration sources (bottom)

There are two source ontologies (Figure 3 - bottom) that are integrated in the above CM. The first source is an online encyclopedia that provides data about different art pieces, offering their title, date of creation, author, used technique etc. Yet rich in content this source is purely text-based. So if one wants to obtain an actual image of a painting we have to consult the second source. This source represents an online multimedia catalogue of exhibits of different kinds including their digitalized versions.

3.2 Integration Model

The integration model (IM) links concepts from the source ontologies (Figure 3 - bottom) to those from the CM: a kind of ontology alignment. Hera provides an integration model ontology (IMO). A designer specifies the links between the CM and the sources by instantiating the IMO. The IMO of Figure 4 describes integration primitives used for ranking the sources within a cluster and for specifying links between them and the CM. The IMO uses the concepts of `Decoration` and `Articulation`. Decorations label the “appropriateness” of different sources (and their concepts) that are grouped within one semantically close cluster. Articulations describe the actual links between the source ontologies and the CM. They also specify the concept’s uniqueness which is necessary to perform joins from several sources.

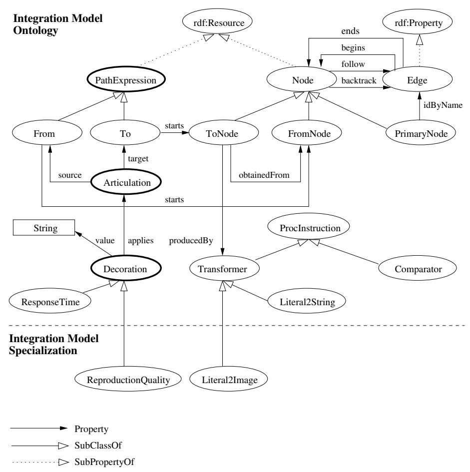

Fig. 4. Integration model ontology and its specialization

3.3 Data Retrieval

Integration is performed once, prior to the user asking the query. Data retrieval is performed for every query. The user query (in RQL [9]) is first extended to contain all relevant data needed in the presentation generation phase. The algorithm traverses the CM from the given concept(s) and adds all concepts and/or literal types reachable by following property edges in the CM graph. Subsequently, the mediator takes the extended query as its input, routes the appropriate queries to the sources, collects the results and assembles them into an answer which consists of a collection of tuples (in RDF terminology a bag of lists). Finally, the “flat” collection of tuples in the query result is transformed into a CM instance by adding the appropriate properties (such that it is a valid RDF graph).

4 Presentation Generation

In the presentation generation phase the designer assembles the retrieved data into a hypermedia presentation suitable for the user's display. Moreover such a presentation is personalized by considering the user preferences for the hypermedia generation.

4.1 Application Model

The Application Model (AM) represents an abstract specification of the presentation in terms of slices and slice properties. A slice is a meaningful presentation unit that groups concept attributes and/or other slices. Each slice is associated to a certain concept from the CM. There are two types of slice properties: slice composition, a slice includes another slice, and slice navigation, a hyperlink abstraction between two slices. The most primitive slices are concept attributes. The most complex ones are the top level slices that correspond to pages. A page contains all the information present on the user's display at a particular moment.

Figure 5 gives an example of an AM composed of two slices associated to two different concepts: `technique` and `painting`. Attributes are depicted with ovals, slice composition is described by the nested composition notation, and slice navigation is represented by arrows. If a slice contains another slice that is associated to a different concept a line labeled with the property name between the involved two concepts is used. In case that the cardinality of this property is one-to-many the `Set` unit needs to be employed.

The application model needs to take in account the user's device (e.g. PC, PDA, WAP phone). To personalize the presentation user preferences are also considered. The user/platform profile captures the device capabilities and the user preferences. The adaptation we consider in this paper is based on the conditional inclusion of slices (fragments) and slice navigation (link) hiding [4]. Figure 5 gives an example of a conditional inclusion of a picture based on the user's display capability to display images.

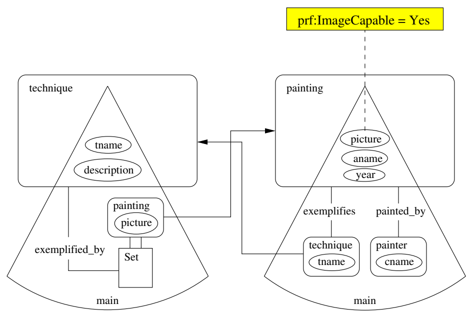

Fig. 5. Application model with adaptation

4.2 Presentation Data Transformations

The retrieved data (CM instance) will go through a number of transformations until it eventually reaches a form interpretable by the user's browser. The RDF/XML serialization of the proposed models and their instances enable the usage of an XSLT processor as the transformation engine. The most important transformations steps are: the application model adaptation, the application model instance generation, and presentation data generation.

After converting the AM to a template closer in form to an AM instance, the AM is adapted based on user/platform profile attribute values. Slices that have their appearance condition unfulfilled will be discarded and links pointing to them will be disabled.

In the second step based on the AM a transformation stylesheet is produced. Since the stylesheet is also an XML document, it was possible to write a stylesheet for the creation of another stylesheet. The resulted transformation stylesheet is used to convert the CM instance to an AM instance.

In the last step different stylesheets produce code for different platforms. This step uses a media-directed translation scheme where for each media appropriate code is generated, e.g. dates are displayed in italic, strings are represented with normal font, or images are refered with appropriate tags. Figure 6 exemplifies the same presentation for three different browsers: HTML, WML, and SMIL browsers. One should note the absence of pictures and the need to scroll the text in the WML browser presentation.

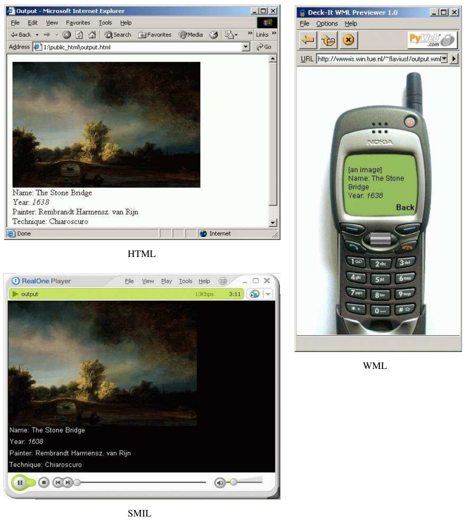

Fig. 6. Hypermedia presentation in different browsers

4.3 User Interaction

There is an increasing number of Web applications (e.g. shopping Web sites, conference management sites, community Web boards) in which the user interaction by only following links does not suffice. In the previous sections we saw the example of user queries that generate a hypermedia presentation. This section describes a more general form of user interaction in which in response to a user click on a button CM instances and their associated properties get created, deleted, or updated.

In order to exemplify the considered user interaction we extend our running example to a shopping Web site selling posters which are reproductions of the museum's paintings. The CM of the application is enlarged with the `poster`, `order`, and `trolley` concepts. A poster is associated (through a property) to the already existing concept of `painting`.

While browsing the generated hypermedia presentation the user has the possibility to order a number of posters corresponding to the currently displayed painting. This implies two modeling issues. First the AM needs to be extended with new navigational properties and possibly new slices that will model this user interaction. Second the callback function associated to this operation needs to be specified based on the current context and the existing hyperbase. The current context is given by the CM instances visible in a hypermedia presentation at a certain moment in time. The callback function is creating, deleting, or updating CM instances and their properties.

The extension of the AM by taking into account the newly introduced concepts modeling user interaction is a trivial process and will not be discussed further. Figure 7 depicts a snapshot of the CM instances during user interaction. The user has put into his trolley, `trolley1`, two orders, `order1` and `order2`. The trolley and the two orders are instances that were created dynamically during user interaction. The first order, `order1`, refers to one copy of `poster2` and the second order refers to two copies of `poster1`. Note that `poster1` and `poster2` are instances that exist independently of the user interaction.

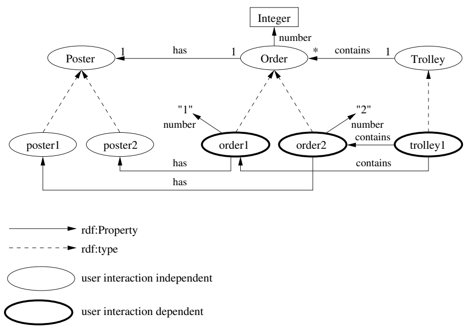

Fig. 7. Concept instance creation during user interaction

5 Conclusion and Further Work

Following the success of the Web, design and engineering methodologies for information systems have evolved in the direction of WIS engineering. Hera's model-driven approach uses Semantic Web technology, most notably RDF(S), to specify the design of a WIS. Its models define integration, data retrieval and presentation generation, but also aspects like user adaptation and user interaction. The choice of RDF/XML to represent the data models and their instances proved to be useful in the data transformations (using XSLT stylesheets). As we develop the Hera methodology by extending it with models for more aspects of the design, for example more advanced user interaction, RDF(S) is not sufficient anymore. Moreover, taking into account the extensions we had to make to RDF(S) we consider replacing it by a more powerful Web ontology language. Other research investigates the consequences in terms of (languages for expressing) transformations on models expressed using Web ontologies.

References

1. Berners-Lee, T.: Weaving the Web. Orion Business, London (1999)
2. Bieber, M., Isakowitz, T., Vitali, F.: Web Information Systems. Communications of the ACM **41(7)** (1998) 78–80
3. Brickley, D., Guha, R.V.: RDF Vocabulary Description Language 1.0: RDF Schema. W3C Working Draft (2002)
4. Brusilovsky, P.: Adaptive Hypermedia. User Modeling and User-Adapted Interaction **11(1/2)** Kluwer Academic Publishers (2001) 87–110
5. Ceri, S., Fraternali, P., Matera, M.: Conceptual Modeling of Data-Intensive Web Applications. IEEE Internet Computing **6(4)** (2002) 20–30
6. Frasincar, F., Houben, G.J., Vdovjak, R.: Specification Framework for Engineering Adaptive Web Applications. In Proc. The Eleventh International World Wide Web Conference (2002)
7. Klapsing, R., Neumann, G.: Applying the Resource Description Framework to Web Engineering. In Proc. Electronic Commerce and Web Technologies, First International Conference, EC-Web 2000, Lecture Notes in Computer Science, Vol. 1875. Springer (2000) 229–238
8. Klyne, G., Reynolds, F., Woodrow, C., Ohto, H.: Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies. W3C Working Draft (2002)
9. Karvounarakis, G., Alexaki, S. Christophides, V., Plexousakis, D., Scholl M.: RQL: A Declarative Query Language for RDF, In Proc. The Eleventh International World Wide Web Conference, ACM Press (2002) 592–603
10. Kay, M.: XSL Transformations (XSLT) Version 2.0. W3C Working Draft (2002)
11. Koch, N., Kraus, A., Hennicker, R.: The Authoring Process of the UML-based Web Engineering Approach. In Proc. First International Workshop on Web-Oriented Software Technology (2001)
12. Lassila, O., Swick, R.R.: Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation (1999)
13. Schwabe, D., Rossi, G.: An Object Oriented Approach to Web-Based Application Design. Theory and Practice of Object Systems **4(4)** (1998) 207–225
14. Wireless Application Group: User Agent Profile Specification. WAP Forum (2001)