Hypermedia Presentation Adaptation on the Semantic Web

Flavius Frasincar and Geert-Jan Houben

Eindhoven University of Technology
PO Box 513, NL-5600 MB Eindhoven, The Netherlands
{flaviusf, houben}@win.tue.nl

Abstract. Web Information Systems (WIS) present up-to-date information on the Web based on data coming from heterogeneous sources. In previous work the Hera methodology was developed to support the design of a WIS. In this paper we target the design of an intelligent WIS. For this reason the Hera methodology is extended with two kinds of hypermedia presentation adaptation: adaptability based on a profile storing device capabilities and user preferences, and adaptivity based on a user model storing the user browsing history. While adaptability is considered to be static, i.e. the presentation is fixed before the browsing starts, adaptivity is dynamic, i.e. the presentation changes while the user is browsing it. The models used in Hera and their adaptation aspects are specified in RDF(S), a flexible Web metadata language designed to support the Semantic Web.

1 Introduction

The Web is the most rapidly growing information source. As huge amounts of data are today stored in the "deep web" (searchable databases), there is an increasing need to automate the presentation of this data. Designed originally for human consumption, the Web is nowadays augmented to target machines. In the Semantic Web [1] era, Web data will evolve from *machine readable* to *machine understandable*, i.e. it will have associated semantics described by its metadata.

The Web can be accessed through a number of different devices (PC, Laptop, WebTV, PDA, WAP phone, WAP watch etc.) each having its own capabilities (display size, memory size, network speed etc.). At the same time, the user preferences (desired layout, navigation patterns etc.) and browsing history can be taken into account during the presentation generation.

Web Information Systems (WIS) [2] offer Web presentations of data typically coming from heterogeneous sources (relational databases, object-oriented databases, XML repositories, WWW etc.). In order to generate an appropriate hypermedia presentation (hyperdocument), the presentation needs to be tailored to specific device capabilities and user preferences.

The Hera methodology [10, 11] supports the design of a WIS. It distinguishes three important design steps: conceptual design that produces the conceptual model of the integrated data, application design that focuses on the navigational or logical aspects of the hypermedia application, and presentation design that gives an abstraction of the physical level of the application. The heart of Hera is the Application Model, a model inspired by Relationship Management Methodology (RMM) [12, 13]. In previous work [9] we built a prototype using the Hera methodology based on XML.

This paper extends Hera by considering the adaptation of the presentation with respect to devices capabilities and user preferences stored in a profile (adaptability). Moreover, we target also the automatic generation of adaptive presentations based on user browsing history stored in a user model (adaptivity). In our methodology the different models lead to a lot of metadata that describe different aspects of the application. Semantic Web technology appears to be a natural solution to represent this metadata. As there is not yet a W3C recommendation for a semantic markup language (only a note on DAML+OIL [5]) we base our future prototype on RDF(S) [3, 15].

The rest of the paper is structured as follows. In Sect. 2 we introduce the Hera methodology and discuss its individual design activities. The artifacts produced by Hera activities are: Conceptual Model, Application Model, and Application Model with Adaptation, presented in Sects. 3, 4, and 5, respectively. Section 5 distinguishes two kinds of adaptations: adaptability described in Subsect. 5.1 and adaptivity described in Subsect. 5.2. Section 6 concludes the paper.

2 Hera Methodology

The Hera methodology is a model-based Web Engineering [16] method for designing WIS. Figure 1 depicts the four different activities of the proposed method: Conceptual Design, Application Design, Adaptation Design, and Presentation Design. The newly introduced activity Adaptation Design is further decomposed in two sub-activities: Adaptability Design and Adaptivity Design.

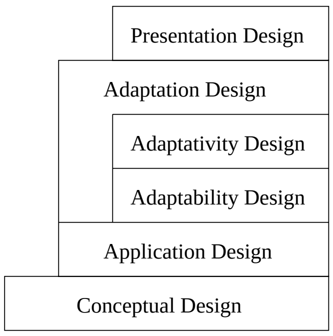

Fig. 1. Hera Methodology

Each activity has specific design concerns and produces a model which is an enrichment of the model built by the previous activity. Hera models are represented in RDFS [3], the schema language for RDF [15]. There are several reasons that motivated us to choose RDF(S). RDFS offers the subclass/subproperty mechanisms useful for building taxonomies for classes/properties. As RDFS is expressed in RDF, it has all the benefits of property-centric models like extensibility and sharability. Extensibility enables the building of each model on top of the previous one and sharability fosters re-use of the developed models. There exist already Composite Capability/Preference Profiles (CC/PP) [14] vocabularies (in RDF(S)) for modeling device capabilities and user preferences which ease the burden of defining new ones.

Conceptual Design provides a common representation for the schema of the retrieved data. A WIS gathers data from multiple sources each having its own dialect. In order to be able to further process this data one needs to define a uniform semantics for it. This semantics is captured in the Conceptual Model (CM) as an application specific ontology. The basic elements in the CM are concepts and concept relationships. Concepts have properties to describe their features.

Application Design is concerned with the navigational aspects involved in the hypermedia presentation of the retrieved data: the structure of the hyperdocument. It extends the CM with navigational views that build the Application Model (AM). The basic elements in AM are slices and slice relationships. Slices are units of presentation for data contained in one or more concepts from CM.

Adaptation Design adds adaptation features to the previously defined AM. We distinguish two kinds of adaptation: adaptability and adaptivity. Both condition the appearance of slices and the visibility of slice relationships. Adaptability does it based on information about device capabilities and user preferences prior to browsing. Adaptivity uses information about the user browsing history stored in a User Model (UM) during the browsing of the presentation. Adaptability is considered to be static, i.e. the presentation is fixed before browsing starts, while adaptivity is dynamic, i.e. the presentation changes while the user is browsing it.

Presentation Design takes into consideration the physical aspects of the presentation. In the Presentation Model (PM) we define slice appearance in terms of regions [10]. The basic elements in PM are regions and region relationships. Regions are rectangular shaped areas that present some data from one or more slices. Slice relationships are materialized by navigational, spatial, or temporal region relationships which can be synchronized. PM is outside the scope of this paper, nevertheless we acknowledge the need of extending the adaptation aspects also to the PM (e.g. font colour, page layout etc.).

3 Conceptual Model

The Conceptual Model (CM) presents a uniform view of the domain semantics for the input data sources. It is an application specific ontology that will constitute the basis for the subsequent data transformations. The retrieved data is so-called "instance data", as it represents specific instances of the concepts defined in the ontology. CM specifies in a hierarchical manner the concepts (classes) in the domain and the relationships among them. These relationships are expressed as class properties (slots). One can associate logic to this ontology that will enable reasoning about CM, e.g. the transitivity of some properties enables the derivation of new properties which can be used at a later stage in our design process.

For CM serialization we were inspired by the RDF(S) representation of ontologies built with Protege-2000 [17]. Protege-2000 is a graphical tool intended for defining ontologies without having to know any ontology language. This graphical representation can be saved in a Protege-extended RDF(S). Figure 2 presents an example of CM that corresponds to a subset of the Rijksmuseum catalogue in Amsterdam.

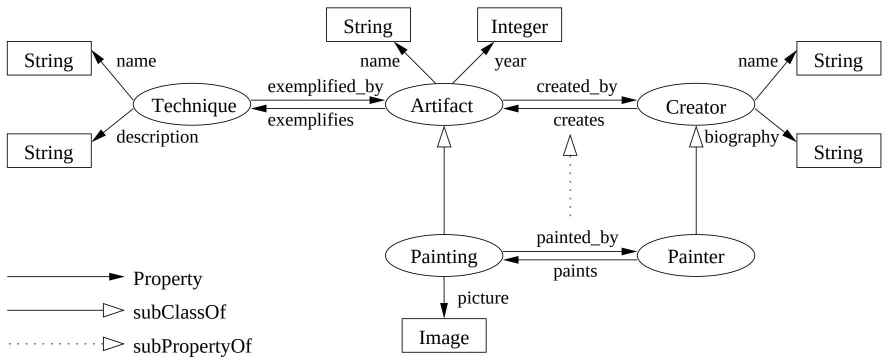

Fig. 2. Conceptual Model

As RDF(S) has its own modeling limitations, we added two new properties to characterize RDF(S) properties that represent relationships: the cardinality (single or multiple) of a relationship and the inverse of a relationship. We also defined a hierarchy of `Media` classes that have specific properties to be used in the Adaptation Design phase. Example 1 shows an instance of the `Image` class: the image dimensions (`pix_x` and `pix_y`) can be considered in adapting the presentation to the display size.

Example 1. Media Type

```
<Image about="http://www.example.com/sunset.jpg"
       pix_x=326
       pix_y=230
       ...
</Image>
```

4 Application Model

The Application Model (AM) describes the navigational view over CM. The AM is the most abstract form of the presentation. We define meaningful presentation units called slices and relationships among them. The simple slices contain only a concept attribute (concept property that points to a media item). Complex slices are defined in a tree-structure manner having simple slices as leaves [12, 13]. We distinguish two kinds of slice relationships: aggregation and reference [9]. Reference relationships are also called links as usually they are materialized to hyperlinks (navigational relationships in the Presentation Model). Each slice belongs to a certain concept from CM. While the reference relationships do not leave the context of a certain concept, aggregation relationships can link slices belonging to different concepts. The designer needs to carefully specify the relationships from the CM, which make such a slice embedding possible. For relationships having cardinality single-many the access structure Set (of slices) is used.

Figure 3 describes a part of the AM for the Rijksmuseum example, i.e. two complex slices `Slice.technique.main` and `Slice.painting.main`.

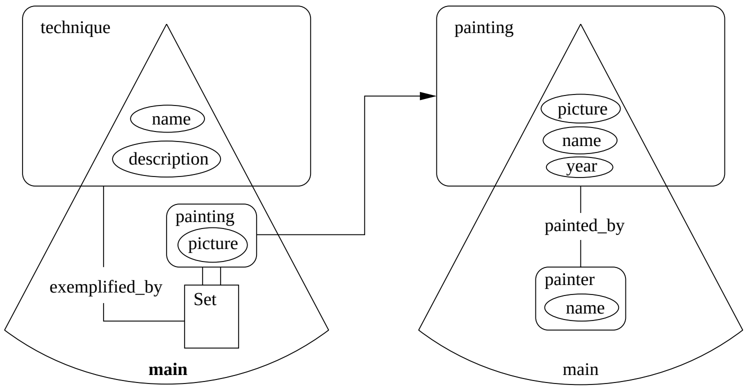

Fig. 3. Application Model

`Slice.technique.main` has two attributes from the concept `technique`, i.e. `name` and `description` and an attribute from `painting`, i.e. `picture`. Since the relationship `exemplified_by` (inherited from CM) has cardinality "single-many" one needs to add an access structure, e.g. `Set`. Note that the previous attributes are slices on their own, i.e. simple slices. Each `picture` has a reference (link) to the `Slice.painting.main` that describes the current painting. This description includes the `picture`, `name`, `year`, and the `name` of the `painter` the painting was `painted_by`. We chose a very simple AM example but rich enough to show, in the next section, different kinds of presentation adaptation.

5 Adaptation in the Application Model

A WIS can be accessed through a multitude of devices and by different users. Each device has its own capabilities (display size, memory size, network speed etc.). Every user has specific preferences (desired layout, navigation patterns etc.) and browsing history with respect to a particular presentation. An intelligent WIS needs to take into account these constraints (abilities) coming from both devices and users, and adapt the presentation accordingly.

The adaptation we consider in this paper is based on conditioning the appearance of slices in the AM (and derived, the visibility of slice relationships or links). Figure 4 shows two examples of slice conditioning, one for adaptability and one for adaptivity. Both examples will be discussed at the level of RDF(S) in the following two subsections.

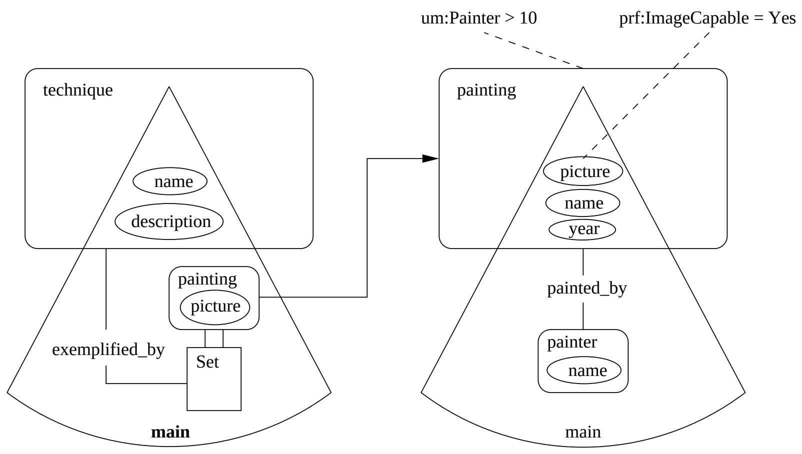

Fig. 4. Adaptation in the Application Model

Example 2 illustrates the definition of the condition for slice appearance in RDFS. A link pointing to a slice that has the appearance condition unfulfilled will be hidden [7].

Example 2. Slice Condition

```
<rdf:Property rdf:ID="condition">
    <rdfs:domain rdf:resource="#Slice"/>
    <rdfs:range rdf:resource="#Literal"/>
</rdf:Property>
```

We consider two kinds of presentation adaptation: *adaptability* and *adaptivity.* Adaptability means conditioning the appearance of slices and the visibility of slice relationships based on device capabilities and user preferences stored in a Profile. Adaptivity takes into account the user browsing history stored in a User Model (UM) to condition the appearance of slices and links during the browsing of the presentation. While we consider adaptability to be static, i.e. the presentation is fixed prior to browsing, we consider adaptivity to be dynamic, i.e. the presentation changes while the user is browsing it.

Two existing techniques fit well in the Hera methodology to model the two kinds of adaptation mentioned above. For adaptability the Composite Capability/Preference Profile (CC/PP) [14] offers a framework to model profiles that characterize device capabilities and user preferences. The AHA (Adaptive Hypermedia Architecture) system [6] adds adaptivity functionality to a hypermedia presentation based on UM.

5.1 Adaptability

Adaptability is the adaptation that considers the device capabilities and user preferences stored in a Profile. A Profile contains attribute-value pairs used by the Hera system to determine the most appropriate presentation for the retrieved data items. Example 3 shows how to build a Profile using two vocabularies, the CC/PP [14] UAProf (User Agent Profile) vocabulary [18] developed by WAP Forum to model device capabilities, and our own vocabulary for describing user preferences.

Example 3. Device/User Profile

```
<Description rdf:about="Profile">
    <ccpp:component>
        <prf:HardwarePlatform>
            <prf:ImageCapable>No</prf:ImageCapable>
            <prf:ScreenSize>600x400</prf:ScreenSize>
            ...
        </prf:HardwarePlatform>
     </ccpp:component>
     <ccpp:component>
        <up:UserPreferences>
            <up:Language>English</up:Language>
            ...
        </up:UserPreferences>
    </ccpp:component>
</Description>
```

A Profile has a number of components, each component grouping a number of attributes. In the previous example we defined two CC/PP components. The `HardwarePlatform` component has two attributes, `ImageCapable` that specifies if the device is able to display images and `ScreenSize` that defines the dimensions of the device display. The `UserPreferences` component has one attribute `Language`, the language the user prefers. Example 4 illustrates an adaptability condition that models the presence of `Slice.painting.picture` in AM based on the ability of the device to display images.

Example 4. Adaptability Condition

```
<rdfs:Class rdf:ID="Slice.painting.picture"
            slice.condition="prf:ImageCapable=Yes">
    <rdfs:subClassOf rdf:resource="#Slice"/>
</rdfs:Class>
```

In an adaptability condition, one can use the media properties defined in Example 1, e.g. one can test if the dimensions of a particular image fit the size of the screen. Note that the adaptation based on user preferences can be treated in the same way as the adaptation based on device capabilities.

5.2 Adaptivity

Adaptivity [4] is the dynamic adaptation that considers the user browsing history stored in a User Model (UM). AHAM (Adaptive Hypermedia Application Model) [8], a Dexter-based reference model for adaptive hypermedia, defines in the Storage Layer three models: the Domain Model, the Teaching Model later on renamed Adaptation Model [19], and the User Model (UM). We use the AHA system to add adaptivity functionality to our methodology. In Hera, the Domain Model will have as atomic concepts slice instances, which stand for both AHAM pages and fragments, and as composite concepts class instances from CM and additional concepts introduced by the application author. The UM contains attribute-value pairs: for each concept from DM a value (from 0 to 100) is associated. The Adaptation Model contains adaptation rules based on the Event-Condition-Action paradigm (like in active databases) that can be executed by the software from the AHA engine.

In Hera when the user visits a certain (page) slice, update rules get triggered that in the end determine the appearance. A slice is desirable if its appearance condition evaluates to true. The desirability of a slice included in another slice implies its appearance in the presentation. Standard AHA functionality implies that a link pointing to a desirable page is displayed in "good" colour ("blue" for active) if it was not visited before, or "neutral" colour ("purple" for visited link) if it was visited before, while "bad" colour ("black" for hidden link) is used if the link points to an undesirable page.

The appearance condition in adaptivity for the slice `Slice.painting.main`, illustrated in Example 5, models the desirability of the slice based on the interest of the user for the painting's painter.

Example 5. Adaptivity Condition

```
<rdfs:Class rdf:ID="Slice.painting.main"
            slice.condition="um:Painter > 10">
    <rdfs:subClassOf rdf:resource="#Slice"/>
</rdfs:Class>
```

Initially, all concepts from UM have their values set to 0. Example 6 shows an update rule expressed in RDF.

Example 6. Update Rule

```
<um:Slice.painting.main>
    <aha:updatelist>
        <aha:SetOfConcepts>
            <aha:item><um:Painting aha:update="+80"/></aha:item>
            <aha:item><um:Painter aha:update="+40"/></aha:item>
        </aha:SetOfConcepts>
    </aha:updatelist>
</um:Slice.painting.main>
```

Suppose that when seeing the slice `Slice.technique.main`, the user is interested in a particular painting description and clicks its link, which for the moment is a link to an undesirable slice (condition not fulfilled), as are all the `Slice.painting.main` instances. Before this slice `Slice.painting.main` is actually visited, the corresponding update rule is triggered. First, the value of the particular instance of `Slice.painting.main` (i.e. associated to the chosen painting) is updated to 35 (default update for a undesirable slice). Then, the values of the Painting and Painter instances from DM (note that these concepts were present also in the CM) corresponding to this particular slice instance have their values updated by `80%` and respectively `40%` of the slice update increment.

As opposed to AHA which specifies update rules for instances, in Hera we specify update rules based on classes (rules are defined at schema level, because the particular instances are not known beforehand). Nevertheless, the Hera update rules will become instance update rules (as in AHA) at runtime.

When after visiting the slice describing a particular painting, the user would go back to `Slice.technique.main`, the user would observe that all the paintings related to the painter of the previously chosen painting are now active links pointing to desirable slices (the condition is fulfilled since the Painter value was updated; $40\% \times 35 = 14 > 10$).

6 Conclusion and Further Work

The Hera methodology has been extended with adaptation models (Profile and User Model) that condition the slice appearance in the Application Model. During the different activities that compose our methodology there are a lot of ontologies involved: Conceptual Model, Domain Model, Profile etc. RDF(S) proves to be a flexible language for modeling, but as we saw, it has also its own shortcomings (we added RDF(S) extensions to cope with them). The Semantic Web promises a richer Web where data will have semantics associated to it. This semantics can be fully exploited in an adapted automatic generation of hypermedia presentations. In the future we plan to include these new emerging Semantic Web standards like DAML+OIL in the modeling of the different ontologies involved in our processes. A prototype based on RDF(S) that experiments with the proposed Hera methodology is under development.

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.: Resource Description Framework (RDF) Schema Specification 1.0. W3C (2000), `http://www.w3.org/TR/rdf-schema`
4. Brusilovsky, P.: Adaptive Hypermedia. User Modeling and User-Adapted Interaction **11(1/2)**, Kluwer Academic Publishers (2001), 87–110
5. Connolly, D., van Harmelen, F., Horrocks, J., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: DAML+OIL (March 2001) Reference Description. W3C (2001), `http://www.w3.org/TR/daml+oil-reference`
6. De Bra, P., Aerts, A., Houben, G.J., Wu, H.: Making General-Purpose Adaptive Hypermedia Work. In Proc. WebNet 2000 World Conference on the WWW and Internet, AACE (2000), 117–123
7. De Bra, P., Brusilovsky, P., Houben, G.J.: Adaptive Hypermedia: From Systems to Frameworks. ACM Computing Surveys, **31(4es)** (1999)
8. De Bra, P., Houben, G.J., Wu, H.: AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In Proc. The 10th ACM Conference on Hypertext and Hypermedia, ACM Press (1999), 147–156
9. Frasincar, F., Houben, G.J.: XML-Based Automatic Web Presentation Generation. In Proc. WebNet 2001 World Conference on the WWW and Internet, AACE (2001), 372–377
10. Frasincar, F., Houben G.J., Vdovjak, R.: An RMM-Based Methodology for Hypermedia Presentation Design. In Proc. Advances in Databases and Information Systems, LNCS 2151, Springer (2001), 323–337
11. Houben, G.J.: Hera: Automatically Generating Hypermedia Front Ends for Ad Hoc Data from Heterogeneous and Legacy Information Systems. In Proc. Third International Workshop on Engineering Federated Information Systems, Aka and IOS Press (2000), 81–88
12. Isakowitz, T., Kamis, A., Koufaris, M.: Extending RMM: Russian Dolls and Hypertext. In Proc. 30th Hawaii International Conference on System Sciences, Computer Society Press (1997)
13. Isakowitz, T., Stohr, E., Balasubramanian, P.: RMM: A Methodology for Structured Hypermedia Design. Communications of the ACM, **38(8)** (1995), 34–44
14. Klyne, G., Reynolds, F., Woodrow, C., Ohto, H.: Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies. W3C (2001), `http://www.w3.org/TR/CCPP-struct-vocab`
15. Lassila, O., Swick, R.R.: Resource Description Framework (RDF) Model and Syntax Specification. W3C (1999), `http://www.w3.org/TR/REC-rdf-syntax`
16. Murugesan, S., Deshpande, Y., Hansen, S., Ginige, A.: Web Engineering: A New Discipline for Web-Based System Development. In Proc. First ICSE Workshop on Web Engineering, ACM Press (1999), 1–9
17. Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Fergerson, R.W., Musen, M.A.: Creating Semantic Web Contents with Protege-2000. IEEE Intelligent Systems, **16(2)** (2001), 60–71
18. Wireless Application Group: User Agent Profile Specification. WAP Forum (1999), `http://www.wapforum.org/what/technical/SPEC-UAProf-19991110.pdf`
19. Wu, H., De Bra, P., Aerts, A., Houben, G.J.: Adaptation Control in Adaptive Hypermedia Systems. In Proc. International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, LNCS 1892, Springer (2000), 250–259