# A DSL Based on CSS for Hypertext Adaptation

Alejandro Montes García, Paul De Bra, George H. L. Fletcher, Mykola Pechenizkiy
Department of Mathematics and Computer Science
Eindhoven University of Technology
Eindhoven, the Netherlands
{a.montes.garcia, p.m.e.d.bra, g.h.l.fletcher, m.pechenizkiy}@tue.nl

## ABSTRACT

Personalization offered by Adaptive Hypermedia and Recommender Systems is effective for tackling the information overload problem. However, the development of Adaptive Web-Based Systems is cumbersome. In order to ease the development of such systems, we propose a language based on CSS to express personalization in web systems that captures current adaptation techniques.

## Categories and Subject Descriptors

H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia

## General Terms

Languages, Theory

## Keywords

Adaptation Language; Adaptive Systems; Personalization

## 1. INTRODUCTION

The web is an enormous hyperspace where users face the problem of information overload. Adaptive web-based systems (AWBS) tackle this problem by displaying only the information that is meaningful for the user. The development of AWBS can be cumbersome and time consuming [4]. Moreover, two kinds of adaptation coexist and are not unified namely, pre-authored (e. g. AHA [10]), and data-driven adaptation (e. g. collaborative recommender systems [2]).

To ease the development of AWBS, we propose a Domain Specific Language [11] (DSL) to express adaptation and only adaptation. This way adaptation is decoupled from the business logic and by doing so, easier integration with existing applications is enabled. The syntax of this DSL must capture the state-of-the art techniques [14] without distinguishing between data-driven and pre-authored adaptation.

We took CSS as a starting point of our DSL because of its usage is widespread in web-based systems and it already defines some rules that can be used for adaptation purposes (e. g. hiding fragments). For this reason it will be easy for CSS developers to develop a fully adaptive system, but also, it will be easy to integrate our language in CSS authoring tools for non proficient CSS developers. This approach also roots our work in the Dexter-based [13] AHAM model [9], that contemplates a layer for presentation specifications.

Adaptation languages for AWBSs already exist [5, 16, 18, 10, 19, 8, 3, 12] but they either do not capture all the adaptation techniques or do not do it in a way that is decoupled from the web system and easy to learn by developers.

## 2. ELEMENTS OF THE LANGUAGE

### 2.1 Selectors

Selectors are the elements of our language that allow the developer to express which DOM nodes will be modified and under which conditions. If we map the DOM nodes to the atomic fragments in AHAM [9] it becomes clear that the adaptation modifies entire DOM nodes.

We distinguish three types of selectors that are useful in a language to express adaptation in AWBS namely, item selectors, user selectors and context selectors. The first ones select nodes from the DOM tree to be modified by the adaptation rules, while the latter ones select to what type of user or in which context the rules will be applied.

An **item selector** is used to select nodes from the DOM tree to be modified by the adaptation rules. CSS defines a set of selectors [6], and their usage is widespread in web systems development, therefore we will use them.

To ease the development as much as we can and develop an approach as close to the standards as possible, we propose **user selectors** (or queries) based on media queries [17]. Media queries are part of the W3C recommendation and their usage is widespread in web development. While those aim to select to which type of device the rules contained inside should be applied, user queries follow the same approach but selecting users instead of devices.

We allow arbitrary features in user queries. The interpreter will search them in the User Model, if those are not found, the query will be evaluated to false and the rules will not be applied. If a feature starts with `min-` or `max-`, its first characters will be deleted and the interpreter will look for the rest and evaluate to true if the value in the UM is higher or lower than the expected. We use `min` and `max` instead of $\leq$ or $\geq$ to stay within the CSS syntax.

It is desirable also to adapt content according to context features. We define **context selectors** the same way as user

*HT'14,* September 1–4, 2014, Santiago, Chile.
ACM 978-1-4503-2954-5/14/09.
http://dx.doi.org/10.1145/2631775.2631782.

selectors. Media queries already do some context adaptation related to device properties. We add the time and location features shown below to provide a wider adaptation effect.

- `min-hour/max-hour`: Hour in which the adaptation will start/stop to be executed.
- `date`: Date in which the adaptation will be executed.
- `min-date/max-date`: Date in which the adaptation will start/stop to be executed.
- `weekday`: Days of the week in which the adaptation will be executed.
- `centre`: A point on the Earth in which the adaptation effect will be provided.
- `radius`: How far a user can be from the previous point.

To connect queries with a logical OR, the user/context queries have to be separated by commas. A logical AND is achieved by nesting them. Different features of one query can also be separated by the keyword `and` to require that every feature is true.

## 2.2 Adaptation Rules

Adaptation rules are used to select the adaptation effects to be provided. For some techniques, standard CSS rules can be used, e. g. hiding fragments. However, for other techniques we have to extend the CSS rules keeping the CSS syntax. Our new rules will be transformed to calls to javascript functions. The following rules have been defined:

- `append(url|HTML)`: Gets the content from `url` or takes it from `HTML` and appends it to the selected node.
- `insert(url|HTML)`: Similar to append, but deleting the previous content.
- `reorder-nodes/reorder-links(strategy)`: Reorders the subnodes (or links contained in an unordered list) according to a strategy (data-driven or expert-based).
- `trim-at(int)`: Trims the text contained in the node at a specified position.
- `fisheye-doi([0.0-1.0])`: Sets the selected value as the value the DOI function has to take in that node.
- `update-attribute(attr,val)`: Sets the attribute named `attr` to be equal to `val` in the selected node.
- `delete-node(bool)`: Deletes the selected node, the boolean indicate whether the content inside the node should be deleted (true) or kept (false).

# 3. USE CASE SCENARIO

To better illustrate our language, let's consider a use case motivated by the CHIP project [1]. CHIP aims to create personalized tours in the Rijksmuseum by using a server-side adaptation engine. Listings 1 and 2 show a possible implementation of some parts of CHIP using our approach.

**Listing 1: adapt.amf**

```
#artworks {
  reorder-nodes: CB-recommender;
}
@user(max-art-knowledge: 30) {
 .description {
  trim-at: 200;
  append: <a>Open description</a>;
 }
 a.advanced {
  delete-tag: false;
 }}
@context(centre:52.36,4.88 and radius:5m){
 a.next {
  update-attribute: href, ./room2.html;
 }}
@context(min-date: 17/03/2014 and max-date
  : 21/03/2014) {
 @context(centre:52.4,4.88 and radius:5m){
  #special {
   display: block;
    append: <p>Picasso exhibition only
       this week</p>;
  }}
 @user(min-surrealism-interest: 50) {
  #upcoming {
   append: <li>Dalí exhibition</li>
  }}}
```

**Listing 2: HTML file from Rijksmuseum**

```
<head>
 <link href="adapt.amf" title="wibaf"/>
 <script src="wibaf.js"/>
</head><body>
 <div style="display:none" id="special"/>
 <div id="artworks">
  <div id="art-cachopo">
   <h2>Cachopo</h2><p class="description">
    This <a href="impress.html" class=
    "advanced">impressionist</a>...</p>
  </div>...</div>
 <ul id="upcoming"><li>Room 3 will be
    renovated soon</li></ul>
 <a class="next" href="./r1.html">Next</a>
</body>
```

Analyzing Listing 1, we see that in the first three lines, the artworks are reordered with the data-driven rule `reorder-nodes`, which is an example of **sorting fragments**.

After that, users with a knowledge in art lower than 30 are targeted. Two adaptation techniques are applied to them. First **stretchtext** is applied to the description of the artworks. After that, links to concepts with the class `advanced` are deleted but the anchor is preserved, i. e. **link hiding**.

Then a case of **guidance** is shown. The current room is detected in and the links to the next room are updated.

The last block is used to display a message if the user is in the right room, saying that there is an special exhibition this week. Then, if the user is interested in surrealism, a list element is appended to the list of upcoming events. Both are examples of the **inserting fragments** technique.

# 4. CONCLUSIONS AND FUTURE WORK

We have demonstrated a CSS-based DSL for adaptation. Our proposed language supports the expression of common data-driven and expert-driven adaptation techniques.

In order to use our language, a Javascript interpreter is being developed at the same time as an adaptive application that uses it. Our interpreter is in an early stage of development but already shows some interesting functionality such as selecting rules according to context and user features.

Future work in this line consists of providing a implementations of methods that perform adaptive operations. Moreover, this language has to be complemented with another one to support the expression of user modeling. In addition, the UM elements should be mapped to ontologies [7] to achieve cross-site integration and enhance personalization [15].

# 5. ACKNOWLEDGEMENTS

The preparation of this article has been supported by the TUE and Adversitement B.V. under the project WiBAF.

# 6. REFERENCES

[1] L. Aroyo, R. Brussee, L. Rutledge, P. Gorgels, N. Stash, and Y. Wang. Personalized museum experience: The Rijksmuseum use case. In J. Trant and D. Bearman, editors, *Museums and the Web 2007*, volume 4511 LNCS, pages 385–389, Toronto, Canada, Apr 2007.

[2] M. Balabanović and Y. Shoham. Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, Mar 1997.

[3] A. Berlanga and F. J. García Peñalvo. Towards reusable adaptive rules. In *Proceedings of the 4th International Workshop on Adaptive Hypermedia and Collaborative Web based Systems*, ICWE 2004, Berlin, Germany, Jul 2004.

[4] P. Brusilovsky, J. Eklund, and E. Schwarz. Web-based education for all: A tool for development adaptive courseware. *Comput. Netw. ISDN Syst.*, 30(1-7):291–300, Apr. 1998.

[5] F. Bry and M. Kraus. Adaptive hypermedia made simple with HTML/XML style sheet selectors. In P. Bra, P. Brusilovsky, and R. Conejo, editors, *Adaptive Hypermedia and Adaptive Web-Based Systems*, volume 2347 of *Lecture Notes in Computer Science*, pages 472–475. Springer Berlin Heidelberg, 2002.

[6] T. Çelik, E. J. Etemad, D. Glazman, and I. Hickson. Selectors Level 3. W3C Recommendation 29 September 2011. `http://www.w3.org/TR/2011/REC-css3-selectors-20110929/`. Accessed: December 12th, 2013.

[7] F. Cena, A. Dattolo, P. Lops, and J. Vassileva. Perspectives in semantic adaptive social web. *ACM Trans. Intell. Syst. Technol.*, 4(4):59:1–59:8, Oct. 2013.

[8] A. Cristea and L. Calvi. The three layers of adaptation granularity. In P. Brusilovsky, A. Corbett, and F. Rosis, editors, *User Modeling 2003*, volume 2702 of *Lecture Notes in Computer Science*, pages 4–14. Springer Berlin Heidelberg, 2003.

[9] P. De Bra, G.-J. Houben, and H. Wu. AHAM: A dexter-based reference model for adaptive hypermedia. In *Proceedings of the 10th ACM Conference on Hypertext and Hypermedia : Returning to Our Diverse Roots: Returning to Our Diverse Roots*, HT' 99, pages 147–156, Darmstadt, Germany, 1999. ACM.

[10] P. De Bra, D. Smits, and N. Stash. The design of AHA! In *Proceedings of the 17th Conference on Hypertext and Hypermedia*, HT '06, pages 171–195, New York, NY, USA, 2006. ACM.

[11] A. V. Deursen, P. Klint, and J. Visser. Domain-specific languages: An annotated bibliography. *ACM SIGPLAN Notices*, 35(6):26–36, Jun 2000.

[12] S. Hagemann and G. Vossen. Web page augmentation with client-side mashups as meta-querying. In N. Nguyen, M. Le, and J. Świątek, editors, *Intelligent Information and Database Systems*, volume 5990 of *Lecture Notes in Computer Science*, pages 23–32. Springer Berlin Heidelberg, 2010.

[13] F. Halasz and M. Schwartz. The dexter hypertext reference model. *Commun. ACM*, 37(2):30–39, Feb. 1994.

[14] E. Knutov, P. De Bra, and M. Pechenizkiy. AH 12 years later: a comprehensive survey of adaptive hypermedia methods and techniques. *New Review of Hypermedia and Multimedia*, 15(1):5–38, 2009.

[15] K. Koidl, O. Conlan, and V. Wade. Towards cross site personalisation. In *Web Intelligence International Conferences*, volume 1 of *WI '13*, pages 542–548, Atlanta, USA, Nov 2013.

[16] M. Kraus, F. Bry, and K. Kitagawa. XML document adaptation queries (XDAQ): An approach to adaptation reasoning using web query languages. In F. Bry, N. Henze, and J. Małuszyński, editors, *Principles and Practice of Semantic Web Reasoning*, volume 2901 of *Lecture Notes in Computer Science*, pages 113–127. Springer Berlin Heidelberg, 2003.

[17] F. Rivoal. Media Queries. W3C Recommendation 19 June 2012. `http://www.w3.org/TR/2012/REC-css3-mediaqueries-20120619/`. Accessed: December 12th, 2013.

[18] D. Smits and P. De Bra. GALE: A highly extensible adaptive hypermedia engine. In *Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia*, HT '11, pages 63–72, New York, NY, USA, 2011. ACM.

[19] N. Stash, A. I. Cristea, and P. De Bra. Adaptation languages as vehicles of explicit intelligence in adaptive hypermedia. *International Journal of Continuing Engineering Education and Life Long Learning*, 17(4):319–336, 2007.