Towards a Model-Driven Product Line for Web systems

Jabier Martinez[1], Cristina Lopez[1], Estibaliz Ulacia[2] and Marta del Hierro[3]

[1] European Software Institute (ESI-Tecnalia)
Parque Tecnológico 204 E-48170 Zamudio, Bizkaia, Spain
jabier.martinez@esi.es, cristina.lopez@esi.es
[2] Ibermática Innovation Institute
Parque Tecnológico 205 E-48170 Zamudio, Bizkaia, Spain
e.ulacia@ibermatica.com
[3] Conmunica Mediatrader
Parque Tecnológico 105 E-48170 Zamudio, Bizkaia, Spain
mdelhierro@conmunica.com

Abstract. Companies focused in the design and implementation of Web based systems, which are also interested in the adoption of state-of-the-art Model-Driven Web Engineering and Software Product Line Engineering techniques, are finding difficulties adapting the requirements of their domains to the already available tooling. This article explains how to successfully combine these two approaches in an industrial context. The techniques described in the case-study show how model-driven product line engineering can be used to provide domain knowledge encapsulation, and automatic generation of the products. These products include both, Content-Manager infrastructures to populate the system's contents, and automatically adapted Web frontend prototype generation for different devices and user profiles.

1 Introduction

The key idea of Software Product Lines (SPL) is the use of a common set of assets for product derivation [1]. In this way, automatic and strategic reuse is achieved through both product commonality exploitation and appropriate variability management. This article presents the industrial usage of a Model-Driven SPL at the Web Systems domain. Product Line Unified Modeller (PLUM) [2, 3, 4] is the SPL Engineering tool suite used to implement the proposed solution for generating a Content-Manager and a Web system prototype adapted for different devices and user profiles.

Industrial companies focused on the development of Web Systems and content providers for Web Systems are trying to improve their daily activities looking for new ways to perform their processes. Apart from a considerable lack of strategic reuse of their Web Systems components, current state-of-the-practice in many of the companies of this market-segment experiment two main problems: On the one hand, the non-existence of a unified way to introduce the contents

derives in an unnecessary waste of time for the employees to learn new technologies and feel comfortable with the new platforms. On the other hand, a rapid prototyping platform is also desirable for showing their customers a working prototype at an early stage.

This article presents a solution for these two problems, achieving at the same time other important goals such as company knowledge encapsulation, automatic and strategic reuse of their components and the benefits of an architecture-driven infrastructure for the company that follows a Model-Driven approach.

Conmunica Mediatrader is a company with expertise in contents, press clipping and product placement. Located in Spain, it belongs to a multimedia leading group in the country. They are specialized in the development of customized applications for their clients. The work described in this paper allows Conmunica Mediatrader to minimize the efforts and costs these developments imply, making their daily work easier.

This paper is structured as follows: Section 2 makes an introduction to the PLUM tool suite that is necessary to understand SPL main assets. Section 3 describes the proposed process to be followed by the company using this approach and sections 4 and 5 describe in detail the implemented models and how the domain variability is interpreted for product generation.

2 Introducing PLUM

PLUM is an open-source tool suite for the design, implementation and management of Product Lines that follows a Model-Driven Software Development approach. PLUM is a domain agnostic tool suite. Its aim is to fill all the needs of different vertical domains with regard to SPL Engineering.

Using PLUM, domain's products variability is captured in what is called a Decision Model. Decision Modeling implies analyzing domain variability in terms of decisions and establishing dependencies among them. This technique, which comes from previous European research projects such as FAMILIES[5], ESAPS[6] and CAFÉ[7], finds in PLUM its latest and more complete implementation up-to-date. The main difference between Feature and Decision Modeling is that in Feature Modeling both domain variability and commonality is modeled while in Decision modeling only the variation points of the domain are modeled. In this way, PLUM differs from other Feature Modeling approaches implemented in commercial tools as pure:variants[8] or Gears[9].

The Decision Model is input for the Variability Resolution Process at which concrete Application models are derived. The Variability Resolution Process consists in giving values to the Decisions in order to define the desired product. In this way Application Models represents concrete products of the product line. This concept is also illustrated in Figures 3 4 and 5 regarding to specific Decision models and Application models of the present case-study.

The transformation of these Application models into the desired output (source code, documentation or other models) is possible through what are called the Flexible Components. These reusable assets interprets domain variability in

order to generate the required products. Another important feature of PLUM consists in Workflow definition. The Workflows establish the process for product generation and deployment. The Flexible components are called from the workflows for code generation, and other deployment processes such as generated code compilation, retrieving dynamic data, launching external processes etc. could be started if needed.

PLUM is integrated in Eclipse[10], diminishing the need for additional development frameworks for final users. It uses a wide range of well established Eclipse technologies not only restricted to the Eclipse Modeling Framework[11]. It also includes GMF[12] for graphical asset editors, EMF Validation framework, OCL[13] for Decision Model's dependency engine, BIRT[14] for reporting valuable SPL metrics, oAW[15] as internal language for model transformations into products' code, ModelBus[16] to allow clients to ask for product generation at a remote server, and SVN[17] for version control system and data source from which historical metrics are obtained.

3 Big picture of the solution

Adoption of SPL architecture and SPL Engineering practices demands a strong company commitment. These initial efforts have to be taken into account in the return on investment (ROI) measurement. The economic aspect is out of the scope of the present article, nevertheless, it is important to notice that company way of working is going to be changed slightly. The company, Conmunica Mediatrader, is intended to adopt an architecture-centric approach that uses a Model-Driven Software Development paradigm. Our proposed Web system lifecycle is separated in two phases, a Generation phase and a Management phase, as shown in Figure 1 and figure 2 respectively. Following subsections aim to explain these implemented phases paying special attention to the first one as is more relevant in the present article context.

3.1 Model-Driven Product generation phase

First product generation sub-phase consists in the requirements gathering with the customer. The main stakeholders reach an agreement with the customer about three main aspects: Web system structure, usability and accesibility requirements and the devices that should be supported. With this information, Web, potential users and device Decision models could be used to derive instances of concrete Application models for each of them. The content architect starts the variability resolution process of the Web model obtaining an instance of the desired Web structure. The usability expert, on their part, makes use of the User application models library looking for user profiles that fits customers expectations of potential users. If a new user profile is needed, he starts the variability resolution process of the User Decision model. Finally, the technical

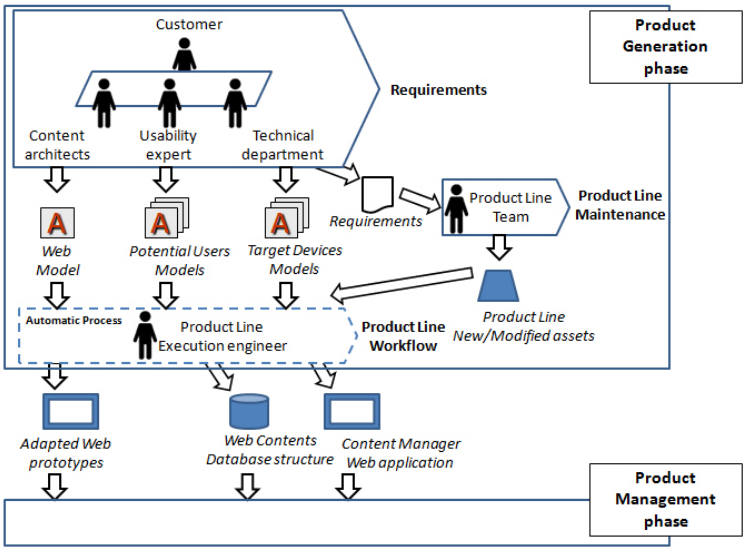

Fig. 1. Product Generation phase

department check if the devices the Web should support are in the Devices Application models library and, also, they start a variability resolution process of the Device Decision model if not found.

PLUM tool suite allows an assisted and guided variability resolution process of the Decision models. In addition, the separation of the different roles allows that the three different Decision models are defined using the domain concepts that each expert understand. For example, the usability expert will decide about potential users possible cognitive impairments while the technical department will decide about connections' quality of service parameters.

As output of the Requirements gathering phase, apart from the application models, the requirements themselves are also inputs for the SPL maintenance team. They are responsible of detecting needs of SPL assets further development. Some requirements could demand supporting new unexpected variability in the decision models, and that could have a direct relation to the flexible components that interprets this variability for Web prototype, database or content manager automatic generation. In summary, requirements interpretation affects SPL assets evolution, in this way the company enrich their reuse capability at the same time that they fill customer needs.

The following phase, the Product Line Workflow, is an automatic process initialized by the PL Execution engineer. Given the Web model defined by the content architect with the customer, a database for storing the Web contents is automatically derived. This is achieved by model-to-text transformations, implemented in the Flexible Components, which takes the Web model and generate a database schema with only the needed tables and columns. The execution of a defined PLUM Workflow allows not only to generate the code of the database, but

also to deploy this code. Using off-the-self Workflow components, the database schema is translated to SQL queries and directly executed in a selected server. In the same manner, the Web Content Manager used for accessing the database is generated. The generated Content Manager is specific for the Web systems domain so its aim is to offer an usable way to introduce Web contents.

Web style, source code language, and how to present the contents vary in terms of user profiles and target devices. The automatic process of variability interpretation through the Product Line assets allows the Adapted Web prototypes generation. The defined Web characteristics are represented in the automatically generated final output obtaining a working Web prototype that could be even shown at very early validation phases.

3.2 Product management phase

Once the products are generated at the previous phase, the Content Manager can be immediately used for contents introduction. However, the adapted Web prototype should go through new defined phases before being in production. The first phase is the validation phase at which the customer or the testers refine Web systems requirements. Their feedback could be very valuable for possible changes in the previously taken decisions.

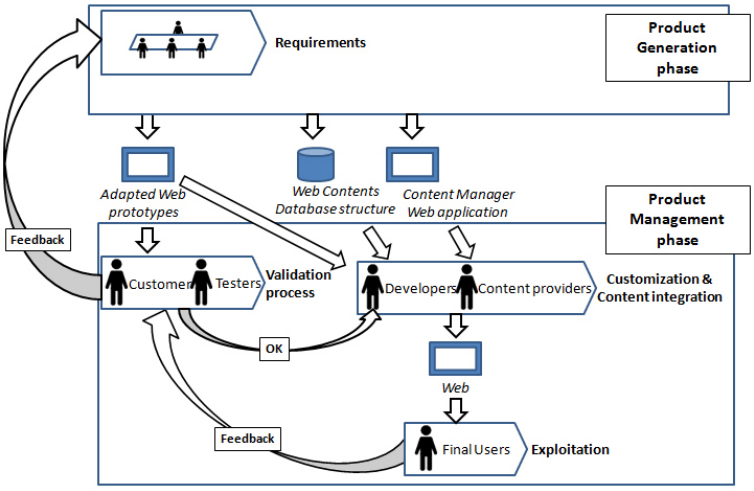

Fig. 2. Product management phase

On the other hand, the generated prototype could be used by the developers to start a incremental prototyping methodology for the development of the Web system. The company deliver to the customer a working Web that could be put online when they consider that this should be done. In order to close the Web system life-cycle, final users could give feedback to the customer that could derive in small code changes or in requirements redefinition.

4 Decision Models assets

In order to model the variability of an environment with user accessibility and multi-device support requirements, we have identified the need of three Decision models to capture the variability of a Web system. The Web Decision model represents the structure and contained information of a Web site. The User Decision model defines user profiles in terms of accessibility aspects. Finally, the Device Decision model represents the main characteristics of the devices that could be used to display the Web system.

The definition of separated models allow us not only to separate the concepts in areas, but also separate its functionality in order to generate tailored solutions. In other words, this makes it possible to create different Web sites adapted for different devices or users maintaining the same Web Application model. It is only needed to change the selected Devices and Users Application models as inputs for executing the adapted Web generation Workflow.

Following subsections explain the implemented Decision models in detail. Due to space restrictions only the most significant decisions of these Decision models are expanded at figures 3, 4 and 5.

4.1 Web Decision model

The Web Decision model has been defined in order to represent the general structure of a Web site. It can be seen as the so-called site map that every single Web site in the Internet has or should have in order to represent its content. The first step when defining a Web, resolving the model variability in a certain Application Model, is to set up a "Name" and a "Description" with the aim of distinguishing the site from others. It is also essential to select the type of multimedia files the site hosts (in case it does), using the choice decision "Multimedia", which possible values are: Audio, Photos, Video and Advertising.

Another important and valuable feature these days is the capability of displaying the content of a Web in several languages. The Web model defined considers this possibility within the group of decisions named "Locations", following the rules for Internationalization and localization [18] used in Computer Science: "Language" implements the ISO 639-2 standard [19] and "Currency" the ISO 4217 standard [20].

The biggest group called "Structure" collects the list of "Categories" that will constitute the site, hence defining the schema or the site map. For each category it is mandatory to define a "Name", the number and type of the sections that it will contain and lastly assigning a "Name" for each section included. Section predefined in the model for this purpose are: News, Events, Contacts, Interviews, Multimedia Galleries, Links, Advertising and Timetable. For each type of section it is also possible to choose between a list of features that could be available or not in the final site. For instance, the section "Event" offers the possibility of including associated locations for an event (which take place in a certain date), associated events, a reference to the source of information or a link to buy tickets online. Besides, each section includes information by default which is

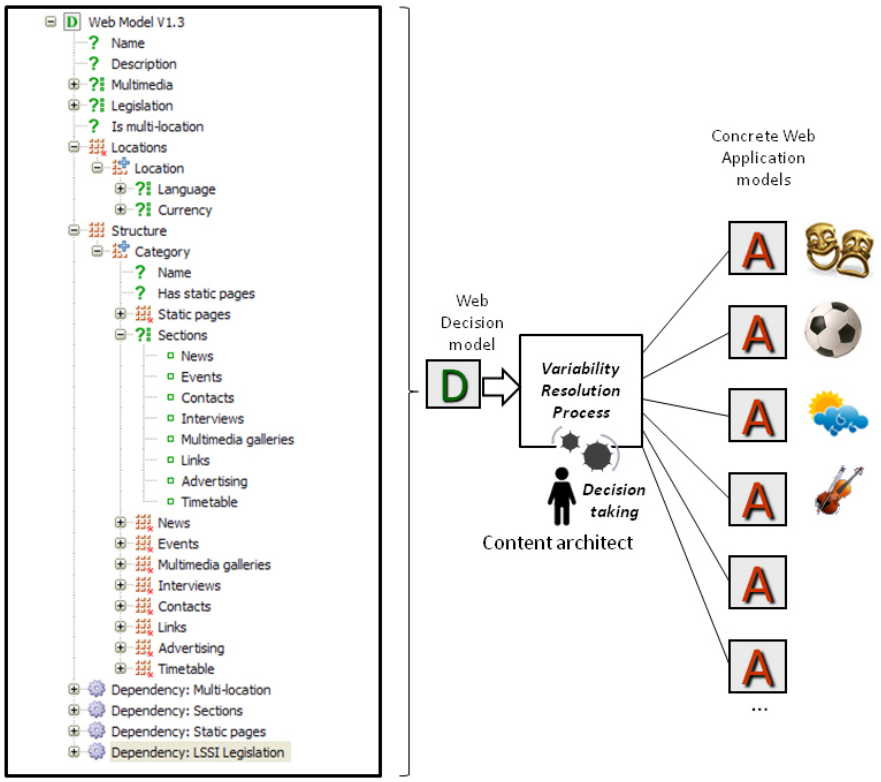

Fig. 3. Web Decision model and examples of derived Application models for different Web sites

always present when defining this type of section, independently of the Web site. Taking this into account, an "Event" section always has a title, a description (divided into paragraphs) and, in case they were selected previously, multimedia files.

The presented solution for Modeling Web Systems does not diverge from the classical Human Computer Interaction engineering approach. The Web System architecture is modeled in terms of a hierarchy of categories and these categories have different sections that vary in its nature. This hierarchy can be seen in other approaches as UIML [21], UsIXML [22] and WebML [23] to name a few. The main difference is that we use higher level descriptions of what a section should have. The natures of the sections were identified during the domain analysis of the company. In this way, our approach does not rely on a set of widgets or abstract interface objects but, as said before, in the nature of each section as a whole.

Another important issue to be considered in web-modeling is User Task Modeling [24]. User Tasks models describes users expected behavior when interacting with the Web system. The Dialog Model [25] is another relevant aspect to be

considered in the model engineering process, it describes the human-computer interaction and how the different dialog states could change. These two models, User Tasks Models and Dialog Models, are omitted from our approach. The reason for that is due to the fact that the knowledge embedded in those two models is encapsulated in the flexible components used to generate the adapted Web Systems. Hence, the company design patterns and best practices for orienting the Web System to different user parameters are encapsulated in the Web System assembly process.

4.2 User Decision model

User modeling is a step that is explicitly declared in many user-centered design processes[26]. The elderly, children and people with special needs, demand different ways of interacting with Web systems and different structures or ways to display the information (the content of the site). Despite of that, a de-facto standard for modeling user profiles, is still missing. For that purpose have arisen initiatives as Discapnet[27], the first Web portal specialized in topics of disability in Spain and pioneer in this kind of studies, promoted by the Foundation ONCE. The accessibility to the Internet from users with special requirements is also a topic promoted by international organizations as the W3C, which have created the Web Accessibility Initiative [28].

The User Model presented in the current article defines a user profile based on the accessibility requirements (in terms of age, experience and disability) that the final user could have at the moment of accessing to a certain Web site. The implementation of this model has been done taking into account one of the requirements defined by Conmunica Mediatrader that consists in the fact that the generated Web sites have to adapt their contents depending on the characteristics or disabilities of the final user. As the Figure 4 depicts, the main features that constitute a user profile are "Age", "Impairment" and "Skill level". These features are considered relevant as they are the main characteristics of a person that could have a strong influence in the way the results are generated or displayed.

The first decision to take in our model, is to select a range for the "Age" feature. Once the range is selected for the Age decision, a set of default values are automatically assigned to the rest of decisions through the defined domain dependencies. For instance, if the value selected is "Elder", it could be taken for granted that the user may have visual impediments, hearing problems, and some kind of cognitive and motor impairment.

The following decision, and probably the most important one, is related with physical impairment or disabilities. The disabilities have been organized in five general groups: visual impairment, cognitive impairment, hearing impairment, motor impairment and language impairment. The values selected in these groups are the ones with most impact on the final Web to be generated. For example , if "low vision", "difficulty using mouse" and "deafness" are selected, then the content in the Web will be generated with big font sizes in both texts and buttons, without scroll bars and multimedia files, such as videos or recordings,

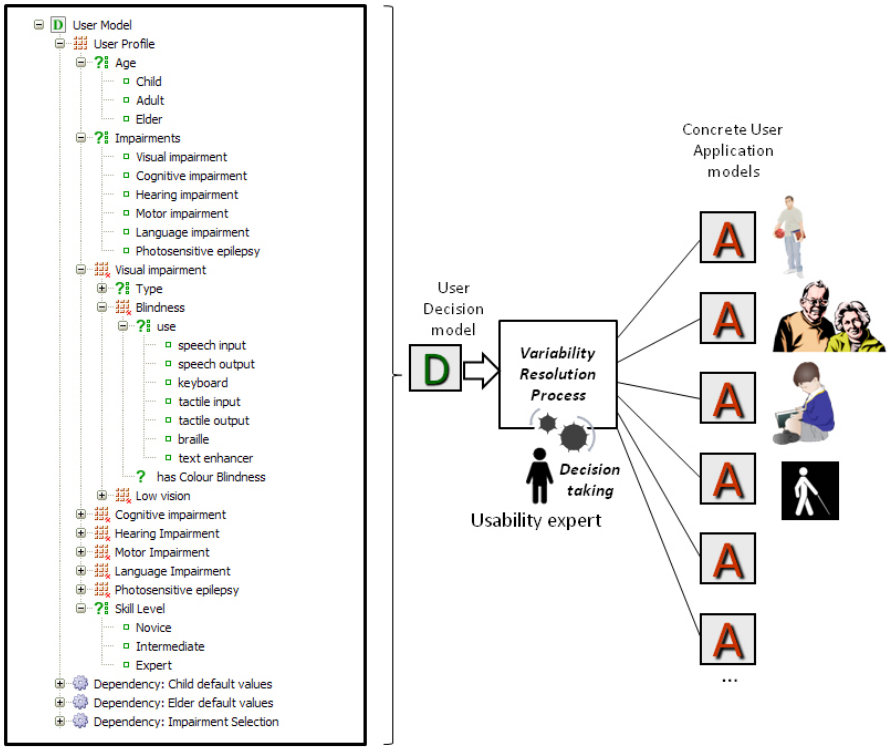

Fig. 4. User Decision model and examples of derived Application models

will be subtitles or omitted. Not only the values selected directly are taken into account but also are the values implicit within each group. Finally, the last decision to make is to choose the skill level of the user. Depending on the level of knowledge of the user, the type of content to be created varies in a great manner. Obviously, a Web Site with specific thematic in which the user has well defined peculiarities differs completely from a general purpose Web Site.

This User Model is based on the Accessibility ontology[29] focused on accessibility semantics and human related characteristics and capabilities. The Human Markup Language [30] is other approach focused on the main characteristics that describe a person and its personality such as gender, belief, culture, emotion, etc. Nevertheless, at the time being, the scope of the case-study only takes into consideration human physical and cognitive related possible impairments.

4.3 Device Decision model

The development teams of Conmmunica are used to develop Web Systems for different device families. This fact has to be considered for the automatic generation of the adapted Web, therefore, a Device Model capable of storing all the device properties is needed. The devices that will be implemented with this

model are mobile devices such as mobile phones or pda's, and Digital Terrestial Television (DTT). Approaches regarding device modeling aimed to User Interfaces adaptation can be found at WURFL [31] for the specific case of mobile devices. Also UsIXML incorporates some entities directly related to device capabilities. Having studied the other approaches, we decided to use the Device ontology specification number SC00091E [32] proposed by the IEEE FIPA Standard committee. This specification comes from the Ambient Intelligence field and it was originally designed for the inter-operation of heterogeneous agents and its services. Despite of that, it fills all of our requirements for device variability knowledge encapsulation.

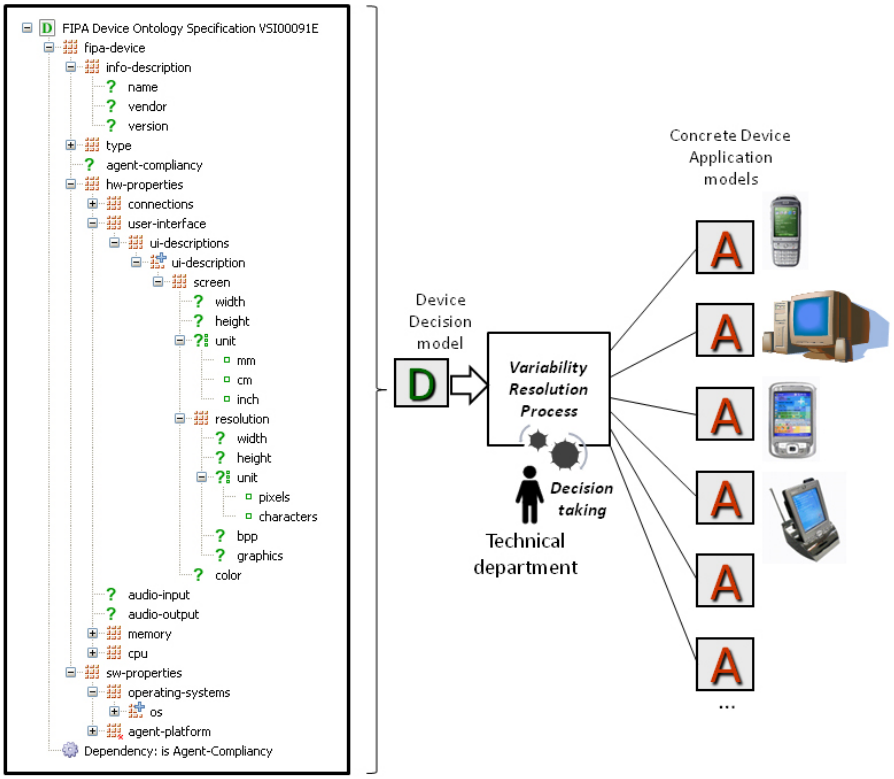

Fig. 5. Device Decision model and examples of derived Application models

Apart from the basic device variable properties such as "name", "vendor" and "version", there are two main groups of decisions: hardware and software properties of the device. Regarding hardware characteristics, we find decisions about supported connections and their quality of service, screens properties, and the availability of audio input or output. There are also decisions regarding memory and cpu properties. On the other hand, the software properties takes

into account the operating systems that the device can support and the agent platforms.

5 Generated products overview

The Flexible Components are the assets that interpret product variability for specific product generation. In this section, in order to illustrate the products generation we present an example. The example is based on a theatre: the "Stacey Theatre".

The content architect defines the Web structure through the variability resolution of the Web Decision Model. The decision making process with the Stacey Theatre owner's requirements allows to obtain a Web application model that defines the desired Web system: The Stacey Theatre Web must have the possibility of being visualized in three languages: English, Spanish and French. Categories and the sections are also defined in the following manner: events and interviews, multimedia galleries of photos and videos, the history of the theatre, contact information, useful links, etc.

The products generated with our solution are a Web site and the corresponding Content Manager application to manage the information contained in the mentioned Web.

5.1 Content manager

The Content manager Web application facilitates the labour of updating the contents of the Web site itself. The automatic generation of this Web system related product represents a competitive advantage for Conmunica Mediatrader. Figure 6 depicts the Content Manager application that is automatically generated from a certain Application Model, more precisely, it shows the Content Manager for the "Stacey Theatre" example. The name displayed at the top is automatically obtained from the name established in the corresponding application model.

The menu on the left side has been fulfilled also automatically from the values available in the "Stacey Theatre" application model, more concisely, from the names of the categories and sections defined within them. Clicking on each item, the application will show the elements that must be fulfilled for the current section and which are totally dependant on the features previously selected for each section in the application model. The application model for the current example defines that the Event section (Plays in the left menu) will not include the "Associated locations", "Associated Events" and "References to sources" elements mentioned in the Web decision model, but it will include a link to buy online tickets. The rest of the elements which are shown in the figure, such as title or description, are the elements considered as mandatory in every single Web site to define an "Event" element, in other words, the minimum information necessary to define the event. For every section defined in the model, it is considered

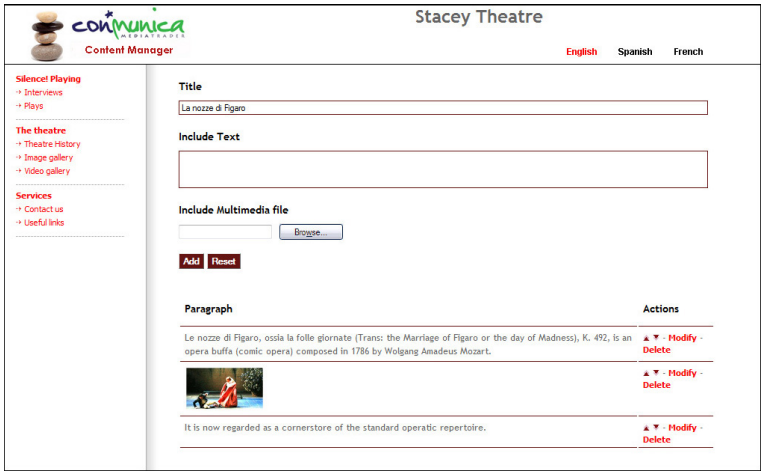

Fig. 6. Screenshot of the Content Manager generated for the Stacey Theatre

that its description is divided into paragraphs, which can be reorganized whenever necessary (with aid of the actions buttons). This implies that is possible to define two kind of paragraphs: plain text and multimedia file (in case the site hosts these files). Finally, all the text data is required in different languages as the links at the top of the main area inform. The languages displayed in these tags depend on the "Locations" defined in the application model, being English, Spanish and French for the "Stacey Theatre" example.

5.2 Adapted Web

The adapted Web obtained from the SPL architecture is not intended to be a Web system ready for production. Despite of that, the obtained Web is a working prototype that allows an instant visualization of any under-design Web system that is being modeled. Apart from that, the generated prototype is adapted using information from specific user and devices models obtaining by this mean a valid approximation of the final result. In adaptable Web Systems research literature, user context and device constraints are taken into account to change not only the Web System but also Service compositions. Using a Product Line approach for achieving these goals is studied at [33]. In that paper, Hallsteinsen et al. present an adaptation platform using also a SPL approach but only taking into consideration device characteristics. We propose the exploitation of both user and device variability to obtain tailored Web systems.

Figure 7 depicts the generated Web for the "Stacey Theatre" example, where it can be appreciated how different the results are depending on the targeted user and device. Specifically, it shows how the Web could change their contents presentation for elderly people, mobile phones and DTT (Digital Terrestial Television).

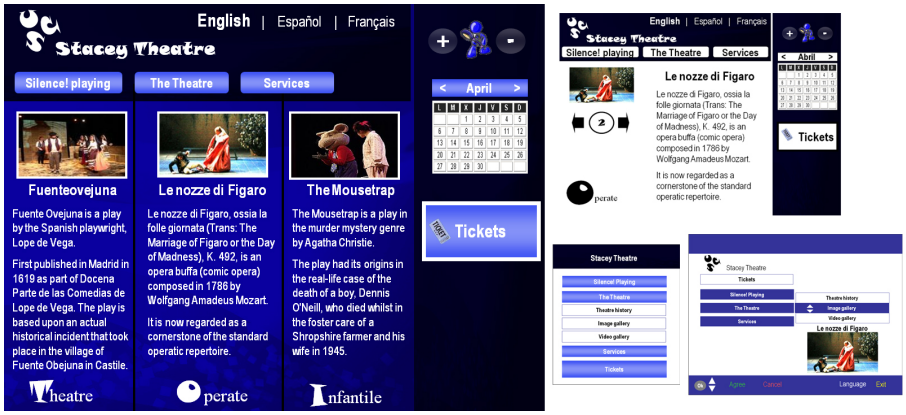

Fig. 7. Different adapted Web prototypes of the Stacey Theatre

With regard to the user adaptation, the most common case of a generated Web is the one in which the output is targeted to someone who does not need a special adaptation. Otherwise, when thinking about adapted Web site, not only we have to focus on people with physical or cognitive disability, but also we have to realize that there are some factors like the age, which determine the way of visualization of certain contents. Because of this, if the generation of a web site for elderly people is required, it is important to consider that they can have some kind of vision impairments or motor impediment, and consequently, it is necessary to provide them with the information adapted to their needs.

Many types of devices can be found nowadays in the market, ranging from devices limited to voice processing to devices with great multimedia processing capabilities. This Device variability is also an important aspect for the Web system adaptation. Supported technologies, multimedia processing capabilities and input/output components determines how the contents are displayed.

6 Conclusions

The current paper presents how to apply Model-Driven Product Line engineering using Decision Modeling techniques in order to capture Web systems domain variability. Our approach accounts for user profiles and devices as they directly affect how concrete Webs are displayed.

We have presented the first results of the Industrial usage of the mentioned approach. The company, Communica Mediatrader, benefits from Content Manager infrastructure and adapted Web frontend prototype automatic generation. These are the Web systems related products obtained from the Product Line implemented with the PLUM tool suite in this case-study.

Acknowledgments

The work reported here has been partially sponsored by the SPRI (Association for the Promotion of Industry) within the Adapta Project (INTEK-BERRI - R&D programs for supporting Basque Country companies) and the "Ministerio de Industria, Turismo y Comercio" within the FLEXI FIT-340005-2007-37 (ITEA2 6022) European project.

References

1. P. Clements and L. Northrop, *Software product lines: practices and patterns*, I. Addison-Wesley Longman Publishing Co., Ed. Addison-Wesley Longman Publishing Co., Inc., 2001.
2. Plum tool suite web. [Online]. Available: http://www.esi.es/plum
3. J. Martinez and J. Vicedo, "Eclipse based variability management tool," in *Eclipse Summit Europe*, 2008.
4. A. Aldazabal and S. Erofeev, "Product line unified modeller (plum)," in *Eclipse Summit Europe*, 2007.
5. Families itea research project. [Online]. Available: http://www.esi.es/Families/
6. Esaps research project. [Online]. Available: http://www.esi.es/esaps/
7. Cafe itea research project. [Online]. Available: http://www.hitech-projects.com/euprojects/cafe/
8. Pure::variants. [Online]. Available: http://www.pure-systems.com/
9. Big lever, gears. [Online]. Available: http://www.biglever.com/
10. Eclipse. [Online]. Available: http://www.eclipse.org/
11. D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks, *Eclipse Modeling Framework*, E. Gamma, L. Nackman, and J. Wiegand, Eds. Addison-Wesley Professional, 2008.
12. Graphical modeling framework. [Online]. Available: http://www.eclipse.org/modeling/gmf/
13. J. Warmer and A. Kleppe, *The Object Constraint Language: Getting Your Models Ready for MDA*. Addison-Wesley Professional, 2003.
14. Business intelligence and reporting tools. [Online]. Available: http://www.eclipse.org/birt/phoenix/
15. Open architecture ware. [Online]. Available: http://www.openarchitectureware.org/
16. Modelbus. [Online]. Available: http://www.modelbus.org
17. Subversion. [Online]. Available: http://subversion.tigris.org/
18. B. Esselink, *A Practical Guide to Localization*. John Benjamins Publishing, 2000.
19. L. of Congress. Iso 639-2. [Online]. Available: http://www.loc.gov/standards/iso639-2/
20. Iso 4217. [Online]. Available: http://www.iso.org/iso/support/faqs/faqs_widely_used_standards/widely_used_standards_other/currency_codes/currency_codes_list-1.htm
21. Uiml (oasis user interface markup language). [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uiml
22. Usixml (user interface extensible markup language). [Online]. Available: http://www.usixml.org

23. Webml. [Online]. Available: http://www.webml.org
24. Paterno, Mancini, and Meniconi, "Concurtasktrees: A diagrammatic notation for specifiying task models," in *Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction*, 1997.
25. Apperley and Duncan, "Human-computer interface design in the software lifecycle," in *Software Education Conference. Proceedings.*, 1994.
26. B. Losada, D. Lopez, and J. Martinez, "Intergram, an user-centered design process," in *9th European Conference for the Advancement of Assistive Technology*, 2007.
27. Discapnet. [Online]. Available: http://www.discapnet.es/
28. Web accessibility initiative. [Online]. Available: http://www.w3.org/WAI/
29. CCCU. Accessonto accessibility requirements repository. [Online]. Available: http://shapevle.cant.ac.uk/AccessOnto/AccessOnto.xml
30. OASIS. (2002) Human markup language. [Online]. Available: http://www.oasis-open.org/committees/download.php/60/HM.Primary-Base-Spec-1.0.html
31. Wurfl (wireless universal resource file). [Online]. Available: http://www.wurfl.com
32. Fipa device ontology specification. [Online]. Available: http://www.fipa.org/specs/fipa00091/SI00091E.html
33. S. Hallsteinsen, E. Stav, A. Solberg, and J. Floch, "Using product line techniques to build adaptive systems," in *10th International Software Product Line Conference (SPLC'06)*, 2006.