Context-driven Progressive Enhancement of Mobile Web Applications: a Multicriteria Decision Making Approach

Heiko Desruelle and Frank Gielen

Department of Information Technology (INTEC) - IBCN, Ghent University - iMinds, Ghent, Belgium
Email: heiko.desruelle@intec.ugent.be

Personal computing has become all about mobile and embedded devices. As a result, the adoption rate of smartphones is rapidly increasing and this trend has set a need for mobile applications to be available at anytime, anywhere, and on any device. Nevertheless the obvious advantages of such immersive mobile applications, software developers are increasingly facing the challenges related to device fragmentation. Current application development solutions are insufficiently prepared for handling the enormous variety of software platforms and hardware characteristics covering the mobile eco-system. As a result, maintaining a viable balance between development costs and market coverage has turned out to be a challenging issue when developing mobile applications. This article proposes a context-aware software platform for the development and delivery of self-adaptive mobile applications over the Web. An adaptive application composition approach is introduced, capable of autonomously bypassing context-related fragmentation issues. This goal is achieved by incorporating and validating the concept of fine-grained progressive application enhancements based on a multi-criteria decision making strategy.

Keywords: Mobile Web applications; Context-awareness; Progressive enhancement; Multicriteria decision making; Adaptive engineering

1. INTRODUCTION

Mobile computing has become a powerful mass medium with a greater reach and faster growth than any other known media type [1]. The sales of mobile devices have recently started to outnumber the traditional laptop and desktop computers. Furthermore, the technology itself is rapidly maturing. Advanced features such as the possibility to install and run third party applications are becoming a standard capability of devices throughout all consumer segments. Through the emergence of installable mobile applications, users are given the freedom to customize and personalize their device as desired. Unlike desktop devices, mobile users carry their device practically at all times. As a result, there is a growing need for mobile applications to become available anywhere, at any time, and on any type of mobile device. However, various technological challenges and limitations reside in the development of such mobile applications [2]. This important limitation is set by the heavily fragmented mobile landscape. Despite numerous bold claims from device manufacturers, the mobile ecosystem is absolutely not all about iPhone devices, nor is it all about Android, Blackberry, or any other platform. Thousands of different mobile devices are currently available, each with their specific characteristics and capabilities, ranging from various operating systems, to different screen sizes, interaction modalities, available APIs, etc. Consequently, the absence of a single mobile application platform makes it mainly the developer's responsibility to resolve fragmentation handling within their application's code [3].

This article focuses on adaptive mobile application development via Web technology. Within this context, the goal of this research is to create optimized mobile Web applications via a progressive enhancement approach. To support the development of such adaptive mobile applications, a method is proposed for extending existing application frameworks with fine-grained mobile progressive enhancement capabilities. Moreover, a novel adaptive application composition algorithm is introduced, which is driven by each user's individual context. The approach aims to cover a

wide range of contextual parameters, covering device capabilities and characteristics (device context), user preferences and profile details (user context), location and time details (environment context). As a result, this process aims to propose a robust and future proof method for the flexible composition of Web applications.

The remainder of this article is structured as follows. Section 2 provides background and related work regarding context-awareness and software adaptability within the Web-based application engineering domain. Section 3 discusses the concepts and algorithmic structure of the adaptive Web application composition approach. Section 4 deals with the architectural aspects of extending application frameworks with such fine-grained progressive enhancement capabilities. Subsequently, Section 5 evaluates the proof-of-concept implementation of this architecture. Furthermore, the evaluation is elaborated based on a use case for implementing an adaptive mobile e-commerce application through the proposed approach. Finally, future work and the conclusion are presented in Section 6.

2. BACKGROUND AND RELATED WORK

As introduced in Section 1, the goal of this research is to provide automated application adaptability for a broad range of mobile devices. This section describes a number of supporting methods as well as alternative approaches for coping with the numerous adaptability requirements in the development process of mobile Web applications.

2.1. The web as an application platform

Developing an application for multiple mobile platforms often requires a skilled and multi-disciplined development team [4]. This requirement considerably drives up an application's development costs and will evidently narrow its target market as well. Against this backdrop, the use of the Web as a generic and cross-platform application solution is rapidly gaining momentum. Device independent Web technologies such as HTML (application structure), CSS (style) and JavaScript (behavior) offer application developers an unprecedented market reach compared to native application development with Java, Objective-C, or C++. Moreover, the International Telecommunication Union (ITU) has estimated the use of mobile Internet connectivity to surpass the access rates of traditional desktop-based Internet by the end of 2013 [5]. Nevertheless, even with the use of standardized Web solutions, efficiently managing mobile fragmentation remains an important research topic. Current Web standards are still primarily agnostic to the variety of available user interaction and presentation modalities. Furthermore, existing mobile Web browsers and runtimes contain many variability points, turning the convergence of mobile applications via Web technology not to be expected any time soon [2]. Recent studies revealed the active use of 2,130 different mobile devices. This study concludes developers currently need to target over 18 devices to reach 50 % of the active mobile market. To reach 60 %, 80 %, or even 90 % of the active mobile users, this number increases to 37, 156, and 330 devices, respectively [6]. In order to reach a sustainable share of the mobile market, Web applications need to be made dynamically adaptable to the contextual environment in which they are being executed. The adaptation and optimization of an application should thus be supported based on contextual parameters such as hardware characteristics, software limitations, the user's profile and preferences, etc. All of this is without the developer's explicit intervention.

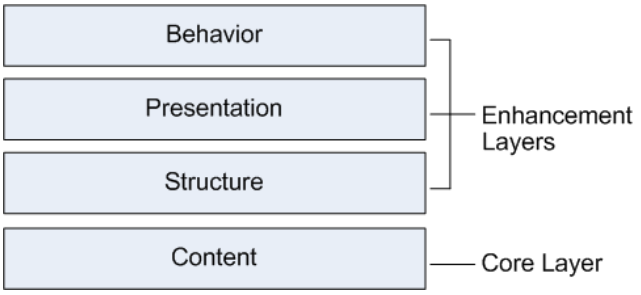

FIGURE 1. Traditional progressive enhancement. A client-side process for incrementally and unobtrusively enhancing a Web application based on a set of specific browser capabilities

Since the early days of Web-based software engineering, developers have tried to cope with the distinct differences between Web browser capabilities. Graceful degradation was one of the first widespread design strategies aiming to do so [7]. The approach focuses on providing optimal support for the most advanced browsers. Less capable browsers are only considered during the very last development stages. Such a design strategy often results in a poor stripped-down version. The graceful degradation methodology expects users to just upgrade their browser when the degraded version does not fit their needs. However, in the context of mobile devices, upgrading the pre-installed browser is in most cases not an option. The default browsers are often an inherent part of the mobile operating system, and alternatives are not always provided.

The progressive enhancement (PE) design approach, on the other hand, reverses the graceful degradation methodology and aims at maximizing usability and accessibility over browsers with different capabilities [8]. Progressive enhancement tries to achieve this goal by forcing developers to take less capable devices into account from the very start of their development process. First, a basic markup document is created, providing an optimal experience for devices with

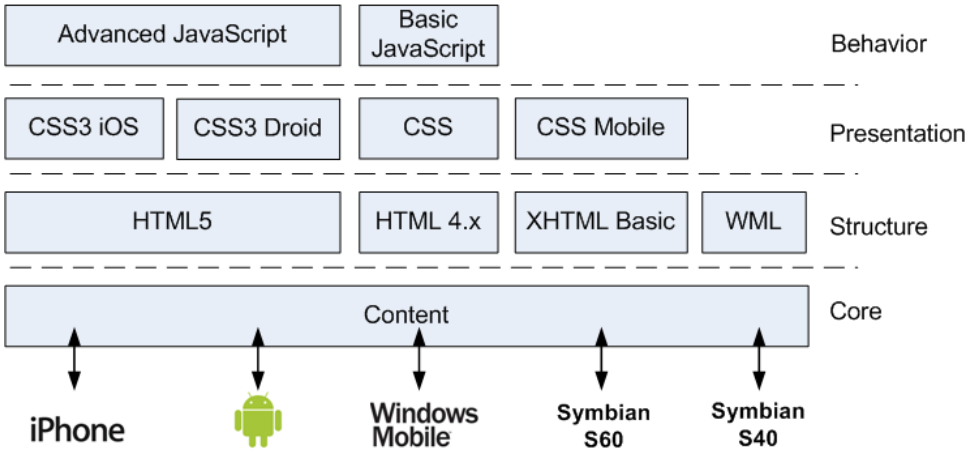

FIGURE 2. Fine-grained mobile progressive enhancement. Dynamic server-side stack composition, driven by the specific characteristics and capabilities of a client's mobile device

the lowest common denominator (LCD) of available capabilities. Incrementally and unobtrusively, one or more layers of structural, presentational, and behavioral enhancements are added to the core application. This step is executed as a function of the requesting browser's specific capabilities. A conceptual overview of the progressive enhancement approach is shown in Figure 1, as it is used for stacking application layers on top of a core content layer.

Progressive enhancement can also be applied in a mobile context to tackle fragmentation related issues. E.g., specific layers can be created to accommodate for various screen sizes. However, when turning this theoretical mobile progressive enhancement approach into actual practice, a considerable number of challenges come into play. Progressive enhancement is a client-side process, executed by the device's browser. The use of CSS3 Media Queries and externally linked CSS and JavaScript resources are the most common practice for selecting an appropriate set of enhancement layers [9]. The number of detectable variability points via this approach is however very limited. As a result, enhancements can only be selected based on the device's screen characteristics and a coarse-grained description of the browser's supported styling and scripting capabilities.

To provide optimized end-user usability, progressive enhancement should also reckon with other contextual parameters, such as the different interaction methods and hardware characteristics offered by mobile devices. E.g., a touch-based device will most likely require some additional presentational enhancement layers, providing a user interface with more space to accurately click buttons, links, etc. In order to set up such a viable mobile progressive enhancement solution, it has become increasingly important to support more fine-grained mechanisms for the applicability evaluation and selection of progressive enhancement layers. As shown in Figure 2, an intelligent mechanism is needed, supporting the automated creation of progressive enhancement stacks based on the user's specific contextual setting.

2.2. Adaptive web engineering

Web-based software systems are traditionally engineered along three orthogonal dimensions: the development phases, the system's views, and its aspects [11]. As illustrated in Figure 3, this approach is characterized by its separation of concerns. The phase dimension sets out the different stages of the Web development process, ranging from analysis, to design, and implementation. Each of these phases requires a number of specific views to be defined, addressing the system's content structures, the navigational structures between content nodes, and its presentation towards the user. Hence, the views dimension. Finally, the aspects dimension sets out the structural and behavioral aspects of each of the above mentioned views.

The growing presence and importance of mobile applications emphasizes the need for fragmentation management within the Web engineering methodology. As identified by Kappel et al., adaptability can be considered as an additional Web engineering dimension, crosscutting all existing Web modeling dimensions [10]. Moreover, Schauerhuber et al. as well as Linaje et al. investigated various modeling methods and found a number of approaches incorporating partial support for adaptability requirements (i.e., UWE, WSDM, HERA, WebML, etc.) [11] [12]. Nevertheless, the applicability of these modeling methods for straightforward and (semi-)automated handling of mobile fragmentation remains rather limited. This observation can be explained through the complex composition of the mobile design space. Various contextual aspects influence a modeling method's expressiveness with regard to adaptability [13]:

- *Context of Use*: In most cases, an application's adaptability requirements are defined in terms of the contextual situation in which it is being executed. As formally defined by ISO 9241-11, the context of use spans a detailed set of descriptions characterizing the application's user, as well as the target device, and the objects in its physical environment [14].
- *Degree of Adaptation*: Adaptability requirements can be static or dynamic. Static adaptation only considers pre-defined versions of an application, specified at design time. Dynamic adaptation, on the other hand, enables applications to be adapted at runtime based on the parameters of their current contextual environment.
- *Adaptation Granularity*: The granularity of an application's adaptability requirements also impacts the engineering complexity. This property indicates whether the defined adaptation process affects the entire application (macro level), or only a contained number of identifiable subcomponents (micro level).

As a result, adaptability requirements are to be handled throughout every stage of an application's development life cycle. Especially from a mobile software development perspective, the multi-dimensional impact of adaptability puts a heavy burden on the application engineering and development processes.

2.3. Mobile context of use modeling

The availability of detailed and reliable metadata regarding a user's contextual setting provides an important driver for enabling a mobile application's dynamic adaptability requirements. The entities represented by this contextual information might influence the adaptability requirements regarding the application's user interface, behavior, content, etc. In initial context-aware research, context was considered to be a component described by two parameters: the end-user's location and the set of objects in his or her immediate vicinity [16]. The subsequent introduction of extensible contextual categories has drastically increased the flexibility of this definition [17]. Chen and Kotz hereto identified five contextual base categories: the device context, the user context, the environment context, the time context, and the historical context [18].

The device context describes the characteristics of the target device that is being used to access the application. The mobile ecosystem covers a wide diversity of screen sizes, interaction methods, software support, etc. In Web-based environments, the device capabilities are generally retrieved through Resource Description Framework (RDF) device profiles, i.e., User Agent Profile (UAProf) and Composite Capability/Preference Profiles (CC/PP) [19] [20]. With this approach, the user's device is identified by matching its user agent string in the HTTP header. In order to facilitate the collection and aggregation of these device profiles, the W3C Mobile Web Initiative (MWI) standardized the Device Description Repository specification (DDR). The specification provides an API and its associated vocabulary for structured access to context providers services via user agent strings [21]. In essence, a DDR thus provides a standardized means for retrieving contextual information about a-priori knowledge on the characteristics of a particular target device or Web runtime. Various open source as well as proprietary DDR implementations are currently actively being maintained. Most notably OpenDDR, WURFL, and DeviceAtlas.

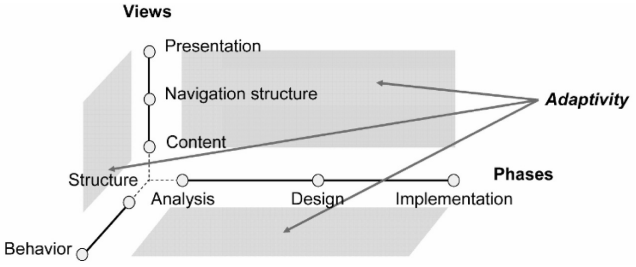

FIGURE 3. Adaptability as a crosscutting aspect on the traditional modeling dimensions of Web engineering (Koch et al. [15])

In a mobile setting, the end-user's profile has gained much importance. Aside from exposing information on user preferences and specific experience, this model should also comprise knowledge regarding the user's specific abilities and disabilities. E.g., enabling accessibility requirements for providing support to elderly people, and people with disabilities. From this perspective, Heckmann proposed the GUMO formalism as a general user model ontology for representing generic user descriptions using the Web Ontology Language semantics (OWL) [22]. The current challenge in this domain is to model the enormous amount of parameters and relationships that characterize the user context [23]. To overcome this issue, forces are being joined with other ontology-driven projects such as Linked Data [24], and UbisWorld [25]. Finally, there are various approaches for aggregating the actual user context knowledge. State of the art reasoners are capable of automatically compose user profiles by examining a user's behavior and data traffic contents (e.g., Google Now, Webinos platform, etc.) [26]. Alternatively, a widely-used approach is to rather focus on aggregating user context knowledge via user preference profiles. In this case, all knowledge is explicitly provided by the user via preference settings. Such a light-weight mechanism obviously reduces the context aggregation system's complexity, but in turn increases the human-computer interface's complexity.

The environment, time, and historical context

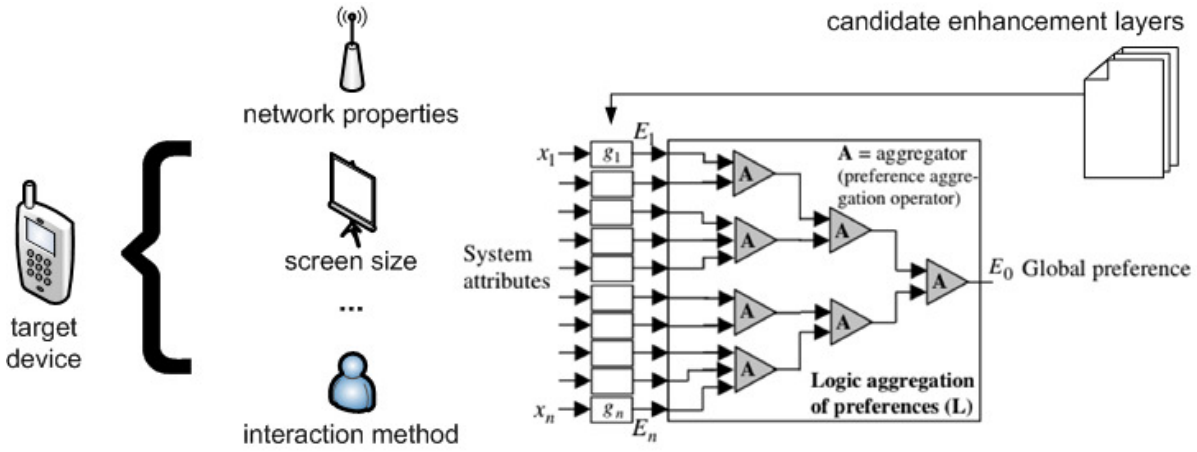

FIGURE 4. The LSP score aggregation mechanism. Each candidate solution is individually matched to a set of pre-defined performance variables. The system is subsequently able to derive the candidate solution's overall suitability score by feeding the resulting elementary degrees of similarity to a logic network of aggregation operators (derived from Dujmovic et al. [29])

are aspects that define where, how, and when the interactions between the user and an application are taking place. The environment context is specified by observing the numerous sensors available on the user's device (e.g., geo-location, temperature, network service discovery, the level of background noise, etc.). Furthermore, the notion of time and historical context is not to be neglected. As context is a dynamic concept, support for temporal pattern recognition and management is needed. The W3C Ubiquitous Web Domain is currently in the process of standardizing the Delivery Context Ontology specification (DCO) [27]. The DCO provides a formal model of the characteristics of the environment in which devices, applications, and services are operating.

3. ADAPTIVE MOBILE APPLICATION COMPOSITION ALGORITHM

This section proposes an adaptive application composition algorithm, enabling the above defined objective to allow for the fine-grained progressive enhancement of Web applications. Mobile applications should provide users with an optimal experience based on the specific contextual capabilities of their device and environment. In order to cope with the wide variety of mobile characteristics, a quantitative evaluation algorithm is introduced, derived from the Logic Scoring of Preference (LSP) method. This adaptive application composition algorithm is designed to support fine-grained progressive enhancement and is capable of suggesting a stack of layers that optimally fits the user's mobile device.

3.1. LSP quantitative evaluation method

The Logic Scoring of Preference method is a quantitative decision method, proposed by Dujmovic [28]. It is designed to assist decision makers in the evaluation, comparison, and selection of complex hardware and software systems. The method has shown its use in various domains where multiple criteria influence the decision making processes. LSP has many applications, especially concerning situations with large and complex solution spaces.

To evaluate a candidate solution, LSP starts by assessing an available solution's similarity with n chosen performance variables. The set of performance variables is denoted as

$$\chi = \{x_1, x_2, \ldots, x_n\}\,. \tag{1}$$

Each performance variable defines a specific property, which an ideal candidate solution is expected to posses. As the algorithm deals with complex decision problems, most candidate solutions will not perfectly match the preset criteria. Nevertheless, such candidates should not be rejected from the very start, as their overall evaluation might still turn out to lead to an acceptable solution. LSP addresses this issue by taking into account how well a candidate matches the different performance variables. For each variable x_i in χ (with $i = 1, \ldots, n$), a corresponding degree of suitability $E_i \in [0, 1]$ is calculated. This score expresses the exact similarity between a candidate solution and the specific performance variable x_i, ranging from 0 to 100 %. In order to attain these scores, LSP requires a predefined mapping function g_{x_i} for each of the n performance variables in χ [29].

$$g_{x_i} : Dom(x_i) \rightarrow [0, 1]\,, \forall x_i \in \chi\,. \tag{2}$$

As defined in Equation (2), the function g_{x_i} maps the degree of similarity between performance variable x_i and the set of values in its corresponding function domain $Dom(x_i)$. Hence,

$$E_i = g_{x_i}(c_i), c_i \in Dom(x_i)\,, \tag{3}$$

TABLE 1. Boolean logic mapping function, leading to value concentration.

Interaction capability	Match
Touch	0 %
Stylus	100 %
Joystick	0 %
Click wheel	0 %

TABLE 2. Fuzzy logic mapping function for optimized value spreading

Interaction capability	Match
Touch	75 %
Stylus	100 %
Joystick	30 %
Click wheel	10 %

by applying the candidate solution's value c_i for performance variable x_i as a parameter to function g_{x_i}. Both Table 1 and Table 2 below illustrate a discrete valued mapping function implementations for the same performance variable in a mobile context. Figure 5, on the other hand, illustrates a continuous mapping function for calculating variable similarities.

After obtaining the elementary degrees of satisfaction, all individual matching scores are to be combined into one objective overall suitability score. This aggregated score is used to determine the best-matching candidate. LSP supports the use of aggregation networks, expressing the mutual relationships between individual scores and how to calculate the overall score (see Figure 4). The standard aggregation operators in LSP are based on the superposition of fundamental Generalized Conjunction Disjunction (GCD) [30]. These operators enable aggregations in terms of partial conjunction, full conjunction, partial disjunction, full disjunction, and neutrality in a single operator. Moreover, a GCD supports the specification of aggregations in terms of 17 graded combinations of conjunction and disjunction. A frequently used implementation for GCDs are Weighted Power Means (WPM), supporting all 17 GCD conjunction/disjunction grades.

$$WPM\left(E_1, E_2, \ldots, E_m; W_1, W_2, \ldots, W_m; r\right) = \left(W_1 E_1^r + W_2 E_2^r + \ldots + W_m E_m^r\right)^{\frac{1}{r}} . \quad (4)$$

The variables W_i in Equation (4) represent the relative weight for each elementary degree of suitability E_i. These weights are mutually balanced by the following requirement:

$$W_1 + \ldots + W_m = 1 . \quad (5)$$

The exponent $r \in \mathbb{R}$, moreover, is determined in function of the aggregation's desired degree of conjunction or disjunction. This flexible exponent allows an evaluator to precisely interlink the mutual

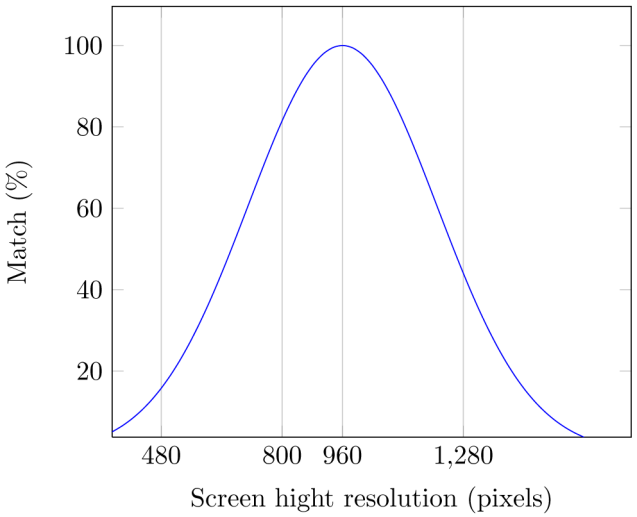

FIGURE 5. Continuous mapping function with value spreading for matching target devices' screen heights

importance of individual suitability degrees within the equation. The calculated aggregation network results in an objective overall suitability score

$$E_0 = L\left(E_1, \ldots, E_n\right) , \quad (6)$$

where the function L defines a logic aggregation network, combining one or more GCDs using a set of individual suitability degrees as input parameters. After calculating E_0 for each of the candidate solutions, conclusions regarding the best-matching solution can subsequently be drawn. The LSP approach selects the candidate with the highest overall suitability score E_0 as the optimal choice for the presented problem.

3.2. LSP selection in a mobile context

The LSP method provides evaluators the ability to flexibly, yet objectively, evaluate complex systems under various contextual circumstances. Such a multicriteria decision making (MCDM) approach can be exploited to implement the envisioned adaptive composition of progressively enhanceable mobile Web applications. In this particular case, a stack of enhancement layers is considered the candidate solution. Each candidate must define the conditions in which it should be able to contribute to an application's optimization, and to what extent the conditions are strictly required or rather considered optional. The specified conditions will be matched to the set of performance variables χ reflecting the client device's supported capabilities and characteristics. The available stacks of progressive enhancement layers are in turn individually evaluated by matching their desired conditions to the mobile user's contextual setting (e.g., the device's available interaction methods, CSS and JavaScript support, etc.). As with the standard

LSP approach, the overall degree of desirability E_0 is expressed in terms of a GCD logic aggregation network. Incorporating the LSP method in a mobile context requires a defined mobile mapping function g_{x_i} for each performance variable x_i. The mapping functions specify the similarity between the target device's performance variables and the specific capabilities supported by the candidate solution (i.e., covered by $Dom(x_i)$, the function domain of x_i).

To illustrate the concept of mobile mapping functions, both Table 1 and Table 2 contain the implementation of a mobile mapping function that compares the performance variable "stylus-based interaction" with a candidate solution's expected interaction method. The function in Table 1 maps the performance variable's domain values using Boolean logic. A Boolean approach implies that only a perfect match is scored. The function implementation in Table 2, on the other hand, uses fuzzy logic and makes much better use of the available scoring interval by also grading the less-than-perfect matches. This second approach is more consistent with the LSP philosophy, as it enables LSP to postpone the final selection decision until all performance variables are evaluated and balanced. The above mentioned examples highlight the importance of carefully thought through mapping functions. In this context, the W3C Mobile Web Best Practices Working Group (MWBP-WG), as well as the W3C Web Accessibility Initiative (WAI) have provided a significant resources repository for the development of mobile LSP mapping functions [31]. The published set of recommendations in the mobile Web usability and accessibility areas are an excellent example of potential sources from which usable mobile mapping functions can be extracted. Figure 5, e.g., depicts a continuous mapping function, matching device's screen resolution with a candidate solution designed for a 960 pixels high screen.

Finally, the elementary scores E_i which resulted from matching the candidate progressive enhancement stacks with the device's specific capabilities are aggregated into an overall suitability score E_0. During this last stage of the selection algorithm, the overall score is attained by applying the candidate's predefined GCD logic aggregation network L. Once all combinations of available candidate solutions have been evaluated, the optimal layer selection process is concluded by selecting the highest scoring progressive enhancement stack.

3.3. Progressive enhancement stack application

Once the optimal stack of progressive enhancement layers has been identified, the final step of the adaptive composition algorithm is to apply the stack's layers to the mobile application. The set of p progressive enhancement layers in the selected stack is applied using a single transformation rule, which can be represented as a generic substitution operation Φ

$$\Phi = \psi\,[l_1/q_1, l_2/q_2, \ldots, l_p/q_p]\ . \tag{7}$$

The transformation Φ in Equation (7) covers p simultaneous substitutions. It consists of an expression ψ which is matched for the patterns l_i that in turn are substituted by their associated expression q_i (with $i = 1, \ldots, p$).

Within the context of applying mobile progressive enhancement layers, ψ represents the lowest common denominator (LCD) version of the mobile application. Each expression q_i represent one of the selected progressive enhancement layers, whilst the associated l_i defines that layer's specific target pattern in the application. For most behavioral and presentational enhancement layers, the pattern to be matched is the Web application's *<head>* tag. Nonetheless, the proposed substitution approach also allows for more complex structural enhancement layers to be supported by matching and transforming specific structural aspects of the application.

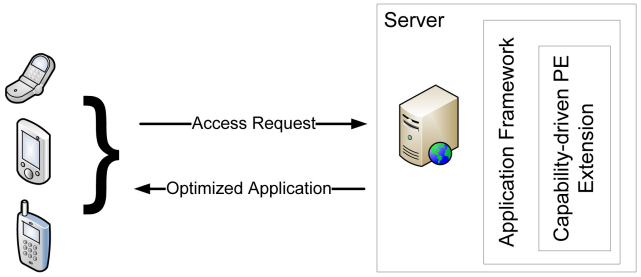

FIGURE 6. A high level system overview. Integrating the fine-grained progressive enhancement approach to optimally serve mobile devices within existing Web application frameworks

4. ARCHITECTURE AND DESIGN

This section discusses the software architecture and data structures needed to integrate the presented adaptive application composition algorithm within an application development environment. First, the algorithm proposed in Section 3 is mapped to an extensible software architecture that selects an optimal stack of progressive enhancement layers based on the client's contextual description and the available enhancement stacks in its repository. Next, the data structure for defining progressive enhancement stacks for this system is elaborated.

4.1. Web application framework extension

In general, developers dedicate substantial efforts in mastering a specific development environment and application framework. From this perspective, it should thus be desirable to support existing frameworks

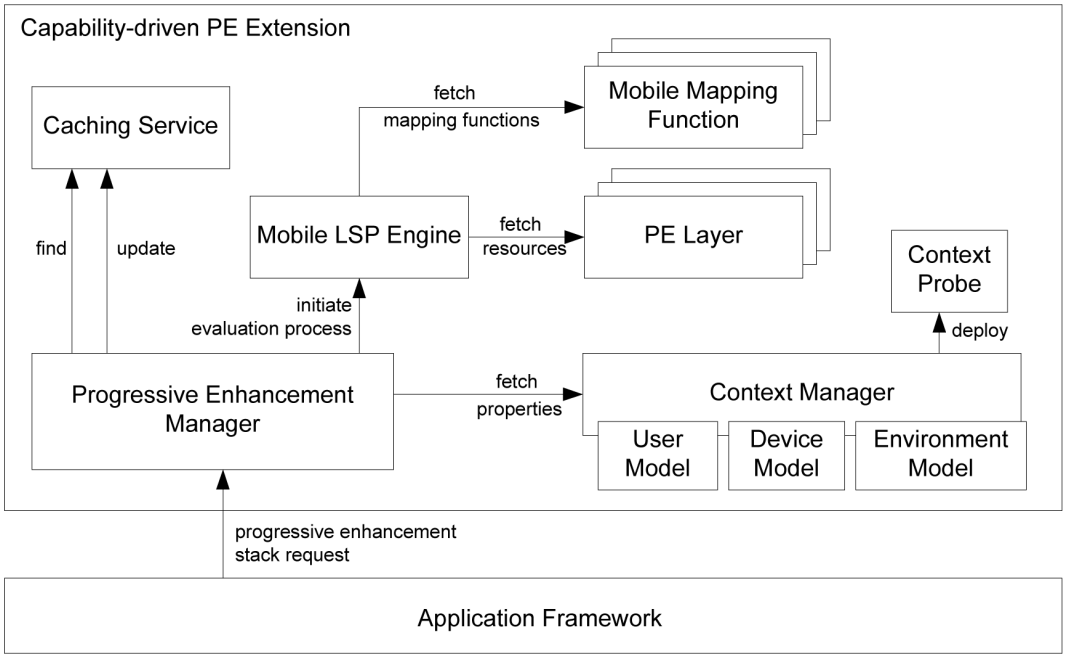

FIGURE 7. Detailed system architecture. Enabling an application framework to request a stack of progressive enhancement layers that optimally suits the user's device characteristics

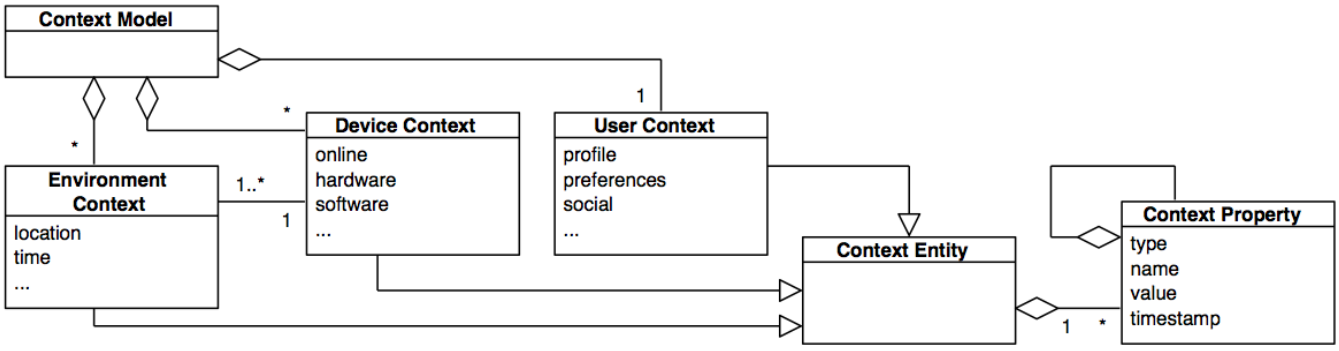

FIGURE 8. Simplified representation of the context sub-models, spanning the mobile context of use

rather than to introduce a completely new development framework. Hence, support for fine-grained mobile progressive enhancement is provided as a generic plug-in for existing Web application platforms. A high level overview of the approach is depicted in Figure 6. The proposed fine-grained progressive enhancement extension interacts with Web application frameworks through a Web interface. Calling this interface selects a progressive enhancement stack that is optimized for the particular capabilities of the current end-user and his device. As a result, application developers are no longer required to manually support the wide range of contextual variability points, as mobile fragmentation issues are handled by the proposed framework extension. What follows is a brief elaboration on the main architectural components of the system, which are shown below in Figure 7.

The Progressive Enhancement Manager represents the system's interface to the outside world. It delegates all incoming requests from the application framework to return an optimized stack of progressive enhancement layers. The manager starts by accessing the Context Manager component, which returns detailed information on the current target device's delivery context. This contextual description spans all contextual knowledge within the system. The model builds upon the W3C's Delivery Context Ontology (DCO) specification [27] and the Context of Use (COU) model proposed by the NEXOF-RA Project [32]. The context of use model comprises three top-level sub-models: the user context, the device context, and the environment context (see Figure 8 for a simplified UML model representation). The models are internally managed and updated by

```
01  <?xml version="1.0"?>
02  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
03
04  <xs:element name="PElayer">
05    <xs:complexType>
06      <xs:all>
07        <xs:element name="preferences" type="LSPnetwork" />
08        <xs:element name="resources" type="enhancements" />
09      </xs:all>
10    </xs:complexType>
11  </xs:element>
12
13  <xs:complexType name="LSPnetwork">
14    <xs:sequence>
15      <xs:element name="preferences" type="LSPnetwork"
16        minOccurs="0"  maxOccurs="unbounded" />
17      <xs:element name="criteria" maxOccurs="unbounded">
18        <xs:attribute name="type" type="xs:string" />
19        <xs:attribute name="value" type="xs:string" />
20      </xs:element>
21    </xs:sequence>
22    <xs:attribute name="aggregationtype" type="xs:string" />
23  </xs:complexType>
24
25  <xs:complexType name="enhancements">
26    <xs:all>
27      <xs:element name="behavior" minOccurs="0">
28        <xs:attribute name="src" type="xs:string" />
29      </xs:element>
30      <xs:element name="presentation" minOccurs="0">
31        <xs:attribute name="src" type="xs:string" />
32      </xs:element>
33      <xs:element name="presentation" minOccurs="0">
34        <xs:attribute name="src" type="xs:string" />
35      </xs:element>
36    </xs:all>
37  </xs:complexType>
38
39  </xs:schema>
```

FIGURE 9. Data structure specification of a mobile progressive enhancement stack using XML Schema Definition (XSD). Each layer defines two major components: (a) the suitability GCD aggregation network, and (b) the stack's enhancement resources (HTML, JavaScript, CSS, etc.)

the system. In order for each of these proposed models to support historical evaluation, pattern detection, and conflict resolution strategies, all stored context properties are timestamped. The contextual information regarding the end-user is described by the user context model. This model consists of an aggregation of user profile data, user preferences, social context information, etc. Furthermore, the device and its physical environment are described by a separate instance of respectively the device context model and the environment context model. The device context model comprises knowledge regarding the corresponding device's hardware characteristics, supported software, etc. The environment context model contains a description of a certain device's location, orientation, etc.

After obtaining all necessary contextual information, the Progressive Enhancement Manager in turn addresses the Caching Service component to find out whether the result for this particular context has previously been calculated and cached. Due to performance considerations, the caching of results for popular devices

is an important part of the system (see performance and scalability evaluation discussion in Section 5). In case of a cache-miss, the execution of the adaptive application composition algorithm triggered. This process is managed by the Mobile LSP Engine component.

The Mobile LSP Engine is responsible for selecting the actual progressive enhancement stack which is to be returned to the application framework. This component is at the heart of the proposed system, as it objectively evaluates the applicability of candidate progressive enhancement layers. The engine starts by fetching all progressive enhancement stacks currently deployed in the system. Next, the mobile mapping functions are retrieved, specifying the degree of similarity between desired context of use parameters and the client's actual situation. The Mobile LSP Engine calculates the overall similarity score for each candidate progressive enhancement stack. Lastly, the engine selects the stack with the highest overall score and thus best matches the characteristics of the client. This final selection is passed back to the Progressive Enhancement Manager component. The manager will cache the mapping between the current delivery context and the resulting enhancement stack. In parallel, the selected enhancement stack is delivered to the requesting application framework.

4.2. Mobile progressive enhancement datastructure

The application composition algorithm relies on a server-side process for the selection of the most suitable progressive enhancement stack. The proposed framework requires a structured description of the available layers, as this data will be used to accurately perform a fine-grained applicability evaluation. To this purpose, an XML Schema Definition (XSD) has been specified, capturing the syntax of a mobile progressive enhancement stack within the system.

As shown in Figure 9, a stack description consists of two major components (specified by the first code block, lines 4-11): a suitability aggregation network (code block A, lines 13-23) and the stack's enhancement resources (code block B, lines 25-37). The aggregation network block specifies a combination of one or more weighted power means and their associated performance variables, which can be freely selected from the available parameters in the context of use model. The aggregation network is needed to enable the LSP stack selection algorithm. The enhancement resources specification block, on the other hand, specifies metadata on the exact type and resource location of this stack's presentational, behavioral, and structural enhancement elements. The enhancement resources are used by the system to correctly identify and apply enhancement layers, once this particular stack has been selected by the LSP selection process.

5. EVALUATION DETAILS

This section evaluates the algorithm and system architecture introduced in Section 3 and Section 4 respectively. First, a proof-of-concept platform implementation is realized. The prototype implementation is subsequently used to validate the adaptive application platform's objectives regarding its usability, performance, and scalability aspects. Two evaluation approaches are applied to validate the usability of the proposed system. For quick and consistent evaluation, a first iteration of usability tests is carried out using an objective automated evaluation mechanism. This iteration is then followed by a round of subjective focus group evaluations for a more detailed validation. Next, the influence of the approach on the application framework's performance and scalability is evaluated through profiling and benchmarking.

5.1. Proof of concept implementation

For the system's prototype, all architectural components have been implemented as part of an extension to the WAFL open source project [33] [34]. The proposed fine-grained progressive enhancement extension has been implemented for both the Drupal [1] and Joomla! [2] Web application frameworks. The platform's context manager was implemented based on the WURFL device description repository [3] (see Section 2.3). In addition, various mobile progressive enhancement layers were created, as a means to validate the system's capability to adapt to the characteristics of heterogeneous delivery environments. The created enhancements range from simple CSS styling layers for feature phones, to complex HTML5/CSS3/JavaScript layers providing a native look-and-feel for high-end smartphones, and even location-aware layers for GPS-enabled devices.

Furthermore, to enable the thorough evaluation of some usability and performance aspects of the system, an adaptive m-commerce application has been built on top of the prototype framework implementations. The commerce application's lowest common denominator (LCD) version provides basic functionality and presentational capabilities. The application LCD enables a consumer to browse through a movie catalog, search for specific entries, and consult product details. Driven by the end-user's context, this experience can be enhanced by enabling the system to automatically load additional PE layers:

- Provide featurephone devices with predictive text input when searching for movie items and enhance it the application's behavior via collapsible list views.

[1] http://www.drupal.org
[2] http://www.joomla.org
[3] http://wurfl.sourceforge.net/

TABLE 3. Focus group evaluation categorization

Group	Device	Characteristics
A	Motorola RAZR	- Lowest common denominator device - Basic HTML structure support - No CSS, JavaScript support
B	Nokia N96	- Mid-range feature phone - Basic HTML structure support - Elementary CSS, JavaScript support
C	Google Galaxy Nexus	- High-end smartphone - HTML5, CSS3, JavaScript support - Android look-and-feel
D	Apple iPhone	- High-end smartphone - HTML5, CSS3, JavaScript support - iOS look-and-feel

(a) (b)

FIGURE 10. Adaptive mCommerce Web application on two feature phones. (a) The Motorola RAZR, a low-end feature phone and (b) the Nokia N96, a mid-range feature phone

(a) (b)

FIGURE 11. Adaptive mCommerce Web application on two smartphones. (a) An Android smartphone, the Google Galaxy Nexus and (b) an iOS smartphone, the Apple iPhone

- Provide Android and iPhone client devices with a mimicked native application look-and-feel.
- Enhance geolocation-enabled devices with directions to nearby store locations.
- Enhance HTML5 notification-enabled client's with information on store promotion via the device's native notification center.

Listing 1. Movie catalog prototype: LCD application skeleton

```
1  <html>
2   <body>
3    <h1 id="title">Products</h1>
4
5    <ul id="items" role="navigation">
6     <li id="bolt" role="item">
7      <a href="bolt.html">
8       Bolt ($10.00)
9      </a>
10    </li>
11
12    <li id="clonewars" role="item">
13     <a href="clonewars.html">
14      Clone Wars ($22.99)
15     </a>
16    </li>
17    ...
18   </ul>
19  </body>
20 </html>
```

As shown in Listing 1, the developer starts by defining a basic LCD version of the movie catalog application. This generic application version ought to work on each targeted device. The application platform, in turn, aims to autonomously find the optimal set of progressive enhancement layers by evaluating their LSP applicability scores. Listing 2, depicts an entry in the system's layer repository. This layer is based on the iUI Web framework [4] and specifically targets iOS devices (lines 14-19). For this purpose, the layer defines a simple full disjunction LSP network matching the target device's context description for an iPhone or iPad model reference, or a Safari mobile browser (lines 1-12). The power of the proposed enhancement approach, however, lies with the mapping functions' fuzzy logic support. As previously noted, this allows less-then-perfect matches to be taken into consideration as well. As a result, e.g., the technologically similar Android browser can fall back to this enhancement in case no specific Android OS enhancement has been deployed in the system (see Table 4).

The remainder of this section will evaluate whether the proposed approach is capable of handling mobile device fragmentation by automatically enhancing the movie catalog application's LCD version in a usable and efficient manner.

[4] https://code.google.com/p/iui/

TABLE 4. Mapping the mobile browser to Safari

Mobile browser	Match
Safari	100 %
Chrome Mobile	75 %
Webkit	70 %
Opera	65 %
IE Mobile	15 %

Listing 2. Movie catalog prototype: iOS enhancement entry from the system's layer repository

```
1  <!-- iOS UI enhancement layer -->
2  <PElayer>
3   <LSPnetwork>
4    <preferences aggr_type="full_disjunct">
5     <criteria type="model_name"
6               value="iPhone" />
7     <criteria type="model_name"
8               value="iPad" />
9     <criteria type="mobile_browser"
10              value="Safari" />
11   </preferences>
12  </LSPnetwork>
13
14  <enhancements>
15   <behavior
16        src="./iui/iui-bootstrap.js" />
17   <presentation
18        src="./iui/iui-iphone.css" />
19  </enhancements>
20 </PElayer>
```

5.2. Usability evaluation

The proof-of-concept m-commerce application is first of all evaluated using a set of automated usability tests provided by the W3C MobileOK test suite service [35]. The MobileOK service checks the usability of Web applications in a mobile context, and is driven by W3C's recommendations and best practices on mobile Web development [36]. The suite of 30 independent tests is based on the validation of markup, application accessibility, content and navigation structuring, load time latencies, and the use of network resources. The prototype m-commerce application gets a perfect score on all MobileOK subtests. Moreover, the application attains a score in the top 10th percentile of all Web-based applications evaluated by this W3C service [37].

The second stage of usability evaluations is carried out using a focus group research methodology. Focus groups are a widely used method for performing qualitative usability evaluations regarding human computer interaction (HCI) [38]. This type of evaluation is designed to obtain a clear perception on the evaluated artifact by running a number of carefully planned discussions with a small group of potential stakeholders [39]. The evaluation interview is conducted in a natural way, where participants are

free to provide their perspective. In this context, test users are categorized according to the type of mobile device they used to evaluate the prototype. As shown in Table 3, four focus groups are held for four different types of mobile users, with devices ranging from low-end feature phones to high-end smartphones. Each focus group has five participants and a moderator guiding the discussion. An additional observer also attends each focus group for aggregating their feedback. During a focus group session, participants are first briefed for 5 to 10 minutes on the context of the system they are about to evaluate. This in order to assure that all participants understand and appreciate the objectives of the approach. Next, the actual evaluation discussion can be initiated and in general lasts for about 30 minutes. As discussed further on, this evaluation shows promising results amongst all test groups.

The user groups operating the devices with limited capabilities (i.e., focus groups A and B) were unanimously positive about the fact that all main application functionality was supported. The application looks and feels optimized for their particular device, without creating the perception of being presented with a stripped-down copy of the smartphone application version. Figure 10(a) shows the basic LCD structure of the m-commerce application, containing only a simple HTML markup. For more capable devices, such as the mid-range feature phones of focus group B, the system detects the applicability of elementary CSS and JavaScript layers. Figure 10(b) depicts this scenario, where the LCD version is automatically enhanced with various presentational as well as behavioral layers.

The feedback from the groups using high-end smartphones (i.e., focus groups C and D) was mainly in the same line. All users recognize the intuitiveness of the provided application. The advanced HTML5, CSS3 and AJAX (Asynchronous JavaScript and XML) support offered by these devices allows the system to select complex enhancement layers. Moreover, a large majority of these groups' participants (80%) were also positive about a Web-based application being able to automatically mimic the underlying operating system's native look-and-feel (see Figure 11(a), and Figure 11(b)). On the other hand, however, there were two users in focus group C raising issues on the interaction fluency of the application. This issue was traced back to the implementation of an AJAX enhancement layer in the system, not the platform itself. The problem was subsequently solved by creating an optimized AJAX enhancement layer for Android devices and deploying it to the system's enhancement repository. After addressing the issue it was no longer identified in one of the later evaluation iterations.

5.3. Performance and scalability evaluation

The adaptive application composition algorithm proposed in Section 3 has a significant influence on the performance of the approach. As the algorithm evaluates the applicability of all available progressive enhancement stack, running time increases linearly with the total number of performance variables to be considered during the selection process of the candidate progressive enhancement stacks. Moreover, if the system contains a repository Γ of n candidate progressive enhancement stacks and each stack $s_i \in \Gamma$ specifies a suitability aggregation network of m_i performance variables to be evaluated, the stack selection algorithm on a single computation thread is expected to consume

$$O\left(\sum_{i=1}^{n} m_i * 2 - 1\right) \tag{8}$$

time for completing the LSP evaluation process. Since the maximum number of nodes in s_i's aggregation network equals $(m_i * 2 - 1)$. In this worst case scenario, all WPMs mean the applicability of only two similarity scores. Nevertheless, the individual evaluations of candidate enhancement stacks are mutually independent and can thus be executed in parallel without considerable overhead. Particularly, the expected computational complexity on a server with τ_p processing threads can be refined to

$$O\left(\max_{i=1..n}(m_i * 2 - 1) * \left[\frac{n}{\tau_p} + \epsilon_{sync}\right]\right). \tag{9}$$

With ϵ_{sync} the thread synchronization overhead. Benchmark and profiling tests on the prototype implementation with Apache JMeter[5] and Xdebug[6] confirm this theoretical prediction. Figure 12 and Table 5 show the average response times of the framework in

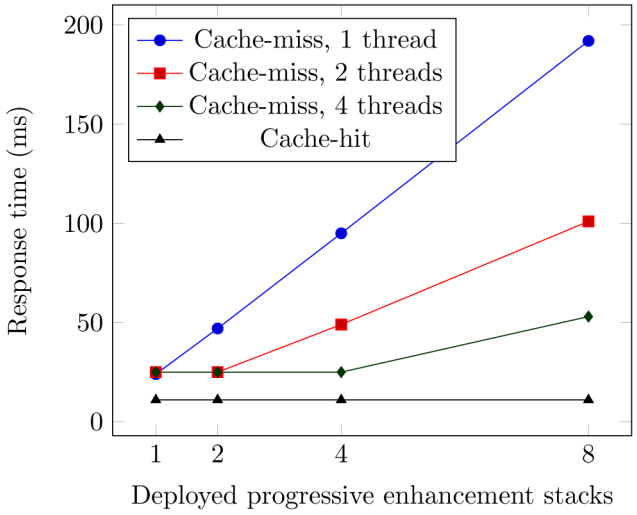

FIGURE 12. System response times for generating optimized progressive enhancement stacks. Measured in function of the number of candidate progressive enhancement stacks in the system

[5] http://jmeter.apache.org
[6] http://www.xdebug.org

TABLE 5. Stack generation response times

Deployed stacks	1	2	4	8
Cache-miss, 1 thread	24ms	47ms	95ms	192ms
Cache-miss, 2 threads	25ms	25ms	49ms	101ms
Cache-miss, 4 threads	25ms	25ms	25ms	53ms
Cache-hit	11ms	11ms	11ms	11ms

function of the total number of performance variables to be evaluated before being able to reach conclusion. Two types of tests are performed. In the first series of tests, a cache-miss is simulated by disabling the system's cache service. This evidently results in a performance degradation due to the algorithm's computational complexity. Because of the algorithm's scalability through parallelization, this effect can be reduced to an acceptable threshold (see Figure 12). For the second series of tests, on the other hand, the system's caching capabilities are enabled. The use of a caching mechanism further improves performance of the system. Results show that in case of a cache hit, the time for selecting an optimal progressive enhancement stack is reduced to a constant execution time. This execution time is regardless of the number of candidate solutions that have to be evaluated by the system.

6. CONCLUSION AND FUTURE WORK

This article introduces a methodology and supporting software framework for the development and delivery of adaptive mobile applications through Web technology. The proposed approach drastically facilitates the development of accessible and usable mobile applications covering a wide range of mobile devices and other contextual parameters. Application developers are no longer solely responsible for manually handling the various aspects of mobile device fragmentation. Only a lowest common denominator (LCD) version of a mobile application needs to be specified. The application presented to the end-user is automatically optimized to the target delivery context through a series of fine-grained progressive enhancements. The proposed adaptive application composition algorithm is based on a multicriteria decision making (MCDM) method derived from the Logic Scoring of Preference (LSP) algorithm. This process is entirely driven by the characteristics of the user's contextual setting, in order to enable an optimal user experience. A proof-of-concept realization was implemented and validated. The extensive end-user usability evaluation of the prototype implementation shows promising results, both for automated as well as focus group validations.

Future work includes the further validation of the proposed approach from the perspective of mobile application developers. From an end-user's point of view, the impact of composing the user context via a more light weight preferences-based approach rather than profile modeling should be investigated. Furthermore, the possible extension of the algorithm towards real time request handling ought to be analyzed. Other research steps in the further development of this adaptive application composition method are related to broadening its scope in terms of supported device types. As the Web is becoming more and more ubiquitous, the diversity of devices that enable access to Web-based applications is extending rapidly. Such devices include tablet computers, home entertainment systems, and even devices from the automotive industry, etc. This evolution further emphasizes the impact of fragmentation handling on application developers. Therefore, the applicability of the proposed application composition algorithm should be evaluated and optimized for more ubiquitous application environments.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7-ICT-2009-5) under grant agreement number 257103 (Webinos project).

REFERENCES

[1] Ahonen, T. (2008) Mobile as 7th of the mass media: cellphone, cameraphone, iPhone, smartphone. Futuretext, London.

[2] Frederick, G.R. and Lal, R. (2009) Beginning smartphone web development. Springer-Verlag, New York.

[3] Desruelle, H. and Lyle, J. and Isenberg, S. and Gielen, F. (2013) On the challenges of building a web-based ubiquitos application platform. Proceedings of the 14th ACM International Conference on Ubiquitous Computing, Pittsburgh, PA, 5–8 September, pp.733–736. ACM, New York.

[4] Cozza, R. and Milanesi, C. and Zimmermann, A. and Glenn, D. and Gupta, A. and De La Vergne, H.J. and Lu, L.K. and Sato, A. and Nguyen, T.H. and Shen, S. (2011) Market Share: Mobile Communication Devices by Region and Country, Gartner inc., Stamford.

[5] International Telecommunication Union (2010) Measuring the Information Society 2010, ITU, Geneva.

[6] Gordon, E.D. (2013) Are Indie App Developers Becoming an Endangered Species? [online]. `http://blog.flurry.com/bid/94811/Are-Indie-App-Developers-Becoming-an-Endangered-Species` [Accessed 8 April 2014].

[7] Florins, M. and Vanderdonckt, J. (2004) Graceful degradation of user interfaces as a design method for multiplatform systems. Proceedings of the 9th International Conference on Intelligent User Interfaces, Funchal, Portugal, 13–16 January, pp. 140–147. ACM, New York.

[8] Wells, J. and Draganova, C. (2007) Progressive enhancement in the real World. Proceedings of the 8th Conference on hypertext and hypermedia, Southampton, UK, 6–11 April, pp. 55–56. ACM, New York.

[9] Lie, H.W. and Celik, T. and Glazman, D. and Van Kesteren, A. (eds) (2010) Media Queries [online]. W3C Candidate Recommendation. Available from: `http://www.w3.org/TR/css3-mediaqueries` [Accessed 8 April 2014].

[10] Kappel, G. and Proll, B. and Retschitzegger, W. and Schwinger, W. (2002) Modeling ubiquitous web applications: the WUML approach. In Olivé, A. (ed), Conceptual Modeling for New Information Systems Technologies. Springer-Verlag, Heidelberg.

[11] Schauerhuber, A. and Wimmer, M. and Schwinger, M. and Kapsammer, E. and Retschitzegger, W. (2007) Aspect-oriented modeling of ubiquitous web applications: the aspectWebML approach. Proceedings of the 14th International Conference and Workshops on the Engineering of Computer-Based Systems, Tucson, AZ, 26–29 March, pp. 569–576. IEEE, New York.

[12] Linaje, M. and Preciado, J.C. and Sanchez-Figuero, F. (2010) Multi-Device Context-Aware RIAs Using a Model-Driven Approach. Journal of Universal Computer Science, **16(15)**, 2038–2059.

[13] Kappel, G. and Proll, B. and Retschitzegger, W. and Schwinger, W. and Hofer, T. (2001) Modeling ubiquitous web applications - A Comparision of Approaches. Proceedings of the 3rd International Conference on Information Integration and Web-based Applications and Services, Linz, Austria, 10–12 September, pp.163–174. Östereichische Computer Gesellschaft, Vienna.

[14] International Organization for Standardization (1998) Ergonomic requirements for office work with visual display terminals (VDT)s - Part 11 Guidance on usability, ISO 9241-11, International Organization for Standardization, Geneva.

[15] Koch, N. and Knapp, A. and Zhang, G. and Baumeister, H. (2008) UML-Based web engineering. In Rossi, G. and Pastor, O. and Schwabe, D. and Olsina, L. (eds), Web engineering: modeling and implementing web applications. Springer-Verlag, London.

[16] Schilit, B. and Adams, N. and Want, R. (1994) Context-aware Computing Applications. Proceedings of the 1st Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, 8–9 December, pp. 85–90. IEEE, New York.

[17] Brown, P.J. and Bovey, J.D. and Chen, X. (1997) Context-aware Applications: From the Laboratory to the Marketplace. IEEE Personal Communications, **4(5)**, 58–64.

[18] Chen, G. and Kotz, D. (2000) A Survey of Context-aware Mobile Computing Research, Technical Report TR2000-381, Dartmouth College, Dept. of Computer Science, New Hampshire.

[19] Open Mobile Alliance (2001) WAG UAProf, Technical Report WAP-248-UAProf-20010530, Open Mobile Alliance.

[20] Kiss, C. (ed) (2010) Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 2.0 [online]. W3C Working Group Note. Available from: `http://www.w3.org/TR/CCPP-struct-vocab2` [Accessed 8 April 2014].

[21] Rabin, J. and Trasatti, A. and Hanrahan R. (eds) (2008) Device Description Repository Core Vocabulary [online]. W3C Working Group Note. Available from: `http://www.w3.org/TR/ddr-core-vocabulary` [Accessed 8 April 2014].

[22] Heckmann, D. (2005) Ubiquitous User Modeling. Thesis (PhD). Dept. of Computer Science, Saarland University, Saarbrucken.

[23] Silva, J.L.T. and Ribeiro, A.M. and Boff, E. and Primo, T. and Viccari, R.M. (2011) A Reference Ontology for Profile Representation in Communities of Practice. Metadata and Semantic Research, **240(1)**, 68–79.

[24] Heath, T. and Bizer, C. (2011) Linked data: Evolving the web into a global data space (1st edition). Morgan & Claypool Publishers, San Rafael, CA.

[25] Heckmann, D. and Loskyll, M. and Math, R. and Rechtenwald, P. and Stahl, C. (2009) Ubisworld 3.0: A Semantic Tool Set for Ubiquitous User Modeling. Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization, Trento, Italy, 22–26, June. Springer-Verlag, Heidelberg.

[26] Desruelle, H. and Isenberg, S. and Lyle, J. and Gielen, F. (2013) Multi-device Application Middleware: Leveraging the Ubiquity of the Web with Webinos. Journal of Supercomputing, **66(1)**, 4–20.

[27] Cantera, J.M. and Rhys L. (eds) (2010) Delivery Context Ontology [online]. W3C Working Group Note. Available from: `http://www.w3.org/TR/dcontology` [Accessed 8 April 2014].

[28] Dujmovic, J.J. (1996) A method for evaluation and selection of complex hardware and software systems. Proceedings of the 22nd International Conference for the Resource Management & Performance Evaluation of Enterprise Computing Systems, San Diego, CA, 10–13 December, pp. 368–378. Computer Measurement Group, Turnersville.

[29] Dujmovic, J.J. and De Tre, G. and Van de Weghe, N. (2010) LSP suitability maps. Soft Computing - A Fusion of Foundations, Methodologies and Applications, **14(5)**, 421–434.

[30] Batyrshin, I. and Kaynak, O. and Rudas, I. (1998) Generalized conjunction and disjunction operations for fuzzy control. Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing, Aachen, Germany, 7–10 September, pp. 52–57. Verlag Mainz, Aachen.

[31] Caldwell, B. and Cooper, M. and Reid, L.G. and Vanderheiden, G. (eds) (2008) Web Content Accessibility Guidelines (WCAG) 2.0 [online]. W3C Recommendation. Available from: `http://www.w3.org/TR/WCAG/` [Accessed 8 April 2014].

[32] Cantera, J.M. and Tsouroulas, N. (2010) Context model and universal APIs [online]. NEXOF-RA Project. Available from: `http://www.nexof-ra.eu` [Accessed 8 April 2014].

[33] Desruelle, H. (2009) Website Adaptation and Formatting Layer [online]. Sourceforge . Available from: `http://wafl.sourceforge.net` [Accessed 8 April 2014].

[34] Desruelle, H. (2010) CMS-driven development of adaptive mobile web applications using progressive enhancement techniques. Thesis (MSc). Dept. of Information Technology, Ghent University, Ghent.

[35] Rabin, J. and Archer, P. (2009) W3C mobileOK Scheme 1.0 [online]. W3C Work Group Note. Available

from: http://www.w3.org/TR/mobileOK [Accessed 8 April 2014].

[36] Connors, A. and Sullivan, B. (eds) (2010) Mobile Web Application Best Practices [online]. W3C Recommendation. Available from: http://www.w3.org/TR/mwabp/ [Accessed 8 April 2014].

[37] W3C (2010) mobileOK usage statistics [online]. W3C mobileOK Checker. Available from: http://www.w3.org/2008/04/mokstats/public/global/20080401-20100110/Overview.html [Accessed 8 April 2014].

[38] O'Donnell, P.J. and Scobie, G. and Baxter, I. (1991) The use of focus groups as an evaluation technique in HCI. People and Computers, **5(1)**, 211–224.

[39] Kontio, J. and Bragge, J. and Lehtola, L. (2008) The focus group method as an empirical tool in software engineering. In Shull, F. and Singer, J. and Sjöberg, D.I.K. (eds), Guide to advanced empirical software engineering. Springer-Verlag, London.