Automatic Generation of Tailored Accessible User Interfaces for Ubiquitous Services

Borja Gamecho, *Student Member, IEEE*, Raúl Miñón, Amaia Aizpurua, Idoia Cearreta, Myriam Arrue, Nestor Garay-Vitoria, and Julio Abascal, *Member, IEEE*

***Abstract*—Egoki is an automatic generator of accessible user interfaces designed to allow people with disabilities to access supportive ubiquitous services. Egoki follows a model-based approach and selects suitable interaction resources and modalities depending on users' capabilities. A proof-of-concept prototype evaluation was conducted with accessibility experts and participants in two different scenarios: one devoted to blind people and the other to people with cognitive impairments. Two services were developed for each scenario: a service for selecting the menu for lunch and a bus schedule service. The tailored user interfaces automatically generated for both scenarios were evaluated by accessibility experts following a barrier walkthrough method. In addition, a user evaluation was performed with eleven participants (two blind participants and nine participants with cognitive impairments). Observational methods were employed during the evaluation sessions where participants were asked to perform several tasks. The results revealed that the tailored user interfaces automatically generated by Egoki were operable and accessible and that all of the participants were able to complete the proposed tasks.**

***Index Terms*—Accessible user interfaces, adaptive systems, automatic user interface generation, tailored user interfaces, ubiquitous computing.**

I. Introduction

THE INREDIS project [1] developed a ubiquitous architecture devoted to supporting people with physical, sensory, and cognitive disabilities. The project's main goal was to grant access to interactive machines to users with disabilities provided with their own mobile device. Its operation was as follows: when a person carrying a mobile device entered a ubiquitous intelligent environment, information about the available local services was displayed on the user's device. If the user requested one of these services, the system downloaded the appropriate user interface (UI) to the mobile device to allow interaction with the selected service. The main outcome of this project was an architecture that provided users with accessible ubiquitous services and a number of demonstrative prototypes [2].

Leveraging the INREDIS project, our Laboratory developed Egoki, a system that automatically generates accessible UIs tailored to people with disabilities, to grant access to ubiquitous services. Ubiquitous computing environments usually provide a single UI for each service. Therefore, people with some specific disabilities can encounter accessibility barriers. To avoid this issue, ubiquitous services' providers ideally should implement different UIs for each group of users who experience access restrictions. However, this solution multiplies the design and maintenance workload. Alternatively, a model-based automatic generation approach has many potential benefits, especially for ubiquitous architectures. Automatic interface generators require a formal description of the service and interface, and the provision of required interaction resources, to generate automatically different UIs tailored to users' needs. In the next section, related work is presented. In Section III, the Egoki system is described. Section IV presents a case study based on evaluating the UIs generated by the system, and Section V presents the conclusions.

Manuscript received March 16, 2014; revised July 15, 2014, September 29, 2014, and November 24, 2014; accepted November 30, 2014. Date of publication January 8, 2015; date of current version September 14, 2015. This work was supported by the Spanish Ministry of Science and Innovation through the ModelAccess project under Grant TIN2010-15549 and by the Department of Education, Universities and Research of the Basque Government under Grant IT395-10. A. Aizpurua, B. Gamecho, and R. Miñón hold Ph.D. scholarships from the latter department. Egokituz Laboratory of HCI for Special Needs is part of the BAILab unit for research and teaching supported by the University of the Basque Country (UFI11/45). This paper was recommended by Associate Editor D. Monekosso.

The authors are with the Department of Computer Architecture and Technology, School of Informatics, UPV/EHU, Donostia-San Sebastián E-20018, Spain (e-mail: borja.gamecho@ehu.es; raul.minon@ehu.es; amaia.aizpurua@ehu.es; idoia.cearreta@ehu.es; myriam.arrue@ehu.es; nestor.garay@ehu.es; julio.abascal@ehu.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/THMS.2014.2384452

II. Recent Work

A. Automatic Generation of User Interfaces

Improved methodologies and tools have been developed in the field of automatic generation of UIs. However, a solution has not yet been reached due to the vast heterogeneity of the elements involved. For the sake of brevity, only a few of those relevant to this work are mentioned here.

Limbourg *et al*. [3] propose the UsiXML language following a paradigm of multipath UI developments. UsiXML allows generating different UIs for different contexts of use. It includes various models at different abstraction levels such as a task model, an abstract UI model, a concrete UI model, and a transformation model. Paternò *et al*. [4] present MARIA, a model-based UI description language for providing support to service-oriented applications, both at design time and at run time. At run time, MARIA supports automatic generation of UIs tailored to different devices. However, user interface markup language (UIML) suits the requirements of Egoki better than UsiXML and MARIA because it provides a mechanism to map different types of resources to each specific interaction element.

UIML has been adopted by diverse UI adaptation approaches. For instance, Feld *et al*. [5] present a UI generator based on it. Similar to Egoki, this work intends to adapt the UI at run time to

apply adaptations such as increasing the margins of the buttons for elderly users.

Kaindl *et al*. [6] explore the semiautomatic generation of UIs for recommendation processes in e-commerce. Egoki has some similarities with this work. Both generate Web interfaces and rely on ontologies. However, while Egoki uses the ontology to store information about the models and separate files for content resources, Kaindl *et al*. use the ontology to store the content information obtained by Web crawling.

ODESeW [7] is an ontology-based application that automatically generates knowledge portals for intranets and extranets. It generates menus according to the user's permissions, visualizing differently the diverse types of information. Similar to Egoki, it uses ontologies to generate UIs that consider involved information. ODESeW focuses on what content should be provided to the users, while Egoki focuses on how to provide the content in an accessible way.

ViMos [8] is a context-aware information visualization service oriented toward providing users with adapted information through devices embedded in the environment. ViMos includes a library of widgets to display multiple types of data by using different visualization procedures and providing adaptive techniques to adjust the visual layout (bearing in mind the content to be displayed and according to the area available in the UI). The main difference is that ViMos generates the UI by means of diverse design patterns, whereas Egoki follows a model-based approach to generating the UI.

B. Adaptation Systems for People With Disabilities

Automated systems for supporting accessibility usually include functions for adapting the UI to the context of use as well as to specific user characteristics stored in user profiles. For instance, the GUIDE system [9] presents adapted UIs for interactive TV oriented toward home services. The MyUI system applies multimodal design patterns [10]; the appropriate design pattern for a given user is selected from her or his user profile. AALuis [11] considers users' physical and cognitive capabilities, their preferences, and their context models, and it applies fuzzy logic to select the modality of the UI. This project is focused on interacting with web services and OSGI services, whereas Egoki interacts with ubiquitous services where the interaction is more infrequent. The Supple system [12] automatically generates standalone UIs adapted to a person's device, tasks, preferences, and abilities. Interface generation is presented as a discrete optimization problem, and a branch-and-bound algorithm using constraint propagation is proposed to solve the problem. Supple generates UIs while considering content, navigation, and presentation adaptations, which are expressed as a cost function. In contrast with Supple, Egoki follows an approach based on model-based UIs.

C. User Interface Adaptation in Other Domains

UI adaptation is a well-established methodology and has influenced the creation of Egoki. Bongartz *et al*. [13] present a system for work environments that generates graphical, vocal, or multimodal UIs adapted to the tasks the user is performing and that considers the user's preferences and expertise. Brusilovsky and Peylo [14] describe several adaptation techniques for learning environments. Their objective is to provide students with different content while considering their needs and taking into account their goals, preferences, and previous knowledge. Some of these techniques for content selection inspired the selection of appropriate interaction resources when designing Egoki.

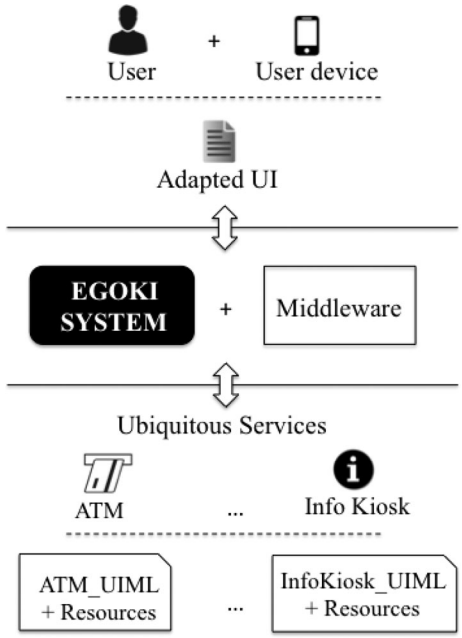

Fig. 1. Example environment for the Egoki system.

III. EGOKI SYSTEM

Egoki is an automated UI generator that creates user-tailored interfaces for accessing services available in ubiquitous environments. Egoki creates UIs starting from models. This paradigm differs from the traditional one consisting of designing a single UI for each service and eventually tailoring it to different groups of users. Egoki requires a logical description of the UI to specify the functionality of that interface. It also requires the provision of suitable multimodal interaction resources. This way, accessible final interfaces can be automatically produced from these formal descriptions, as proposed by the Cameleon Reference Framework [15]. Accessibility features drive the UI generation process to ensure the UI's suitability for users with disabilities.

A. Egoki Ubiquitous Environment

The application scenario of the Egoki system is composed of three types of elements: ubiquitous services provided by local automated machines, such as ATMs, Information kiosks, and vending machines; a middleware layer that manages low-level interactions; and a number of mobile user devices. Fig. 1 shows an example of a ubiquitous scenario for Egoki.

In this scenario, ubiquitous services are provided by applications running on local automated machines. This way, these

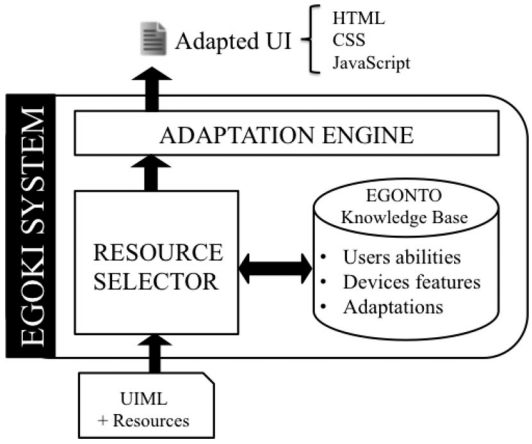

Fig. 2. Architecture of the Egoki system.

services can be made available to mobile devices located within the local wireless network range.

These machines are usually interconnected through heterogeneous wired and/or wireless networks. In addition, the services provided are very diverse; they are designed by different manufacturers, provided by different vendors and run on different devices. Therefore, a middleware layer is required to ensure interoperability among them.

The middleware layer interconnects the diverse networks using different protocols, manages communications between devices, and integrates incoming mobile devices to the system by means of "discovery" mechanisms. The middleware layer provides homogeneous methods to read and modify the state of the ubiquitous services.

We adopted the Universal Control Hub (UCH) [16] to act as a middleware layer. Its implementation is based on the Universal Remote Console (URC) Standard [17]. This standard provides every user device with a unified access point to machines and services available in a network by means of a mechanism called User Interface Socket (UIS). The UIS describes the functionalities of devices and services in terms of abstract UIs. Interfaces generated by Egoki only use the UIS to manage the interaction with ubiquitous services, as described in Section III-C5.

B. Architecture of the Egoki System

The Egoki system is composed of three modules: knowledge base (KB), resource selector (RS), and adaptation engine (AE). These interact with each other to process the inputs of the system and generate the corresponding user-tailored interface (see Fig. 2).

The KB module uses an ontology called *Egonto* to store, update, and maintain the models regarding user abilities, access device features, and interface adaptations.

The RS module queries the ontology to obtain the user's abilities and the device features so that it can select appropriate resources for each user, mobile device, and service. It requires as inputs a logical specification of the UI and different types of interaction resources.

The AE module performs necessary transformations and applies required adaptation rules to generate the final accessible and user-tailored interface. More details about Egoki's architecture can be found in [18].

C. Models

The Egoki system includes diverse models for the characterization of users, devices and types of adaptation, and a number of reasoning rules. All have been defined in the *Egonto* ontology. This information is accessed through the system's KB. The Protégé ontology editor has been used for the creation of these models. The Pellet [19] reasoner has been used for accessing the information from the KB, storing new data, and inferring information about the required adaptations. In addition, a UI model has been defined. It describes the functionalities of the ubiquitous service to be adapted that are used by the Egoki system.

1) User Model: The user model defined by the *Egonto* ontology is based on the interaction abilities of each user. Thus, appropriate resources can be selected to generate a user-tailored interface. In particular, four general interaction abilities and several specific ones are included. Their corresponding ontology classes are depicted in Fig. 3, and the general interaction abilities are described below.

1) *Cognitive:* Skills involved in human communication and in the interaction with technological devices: attention, concentration, concept formation, language processing, learning, memory, and perception [20].
2) *Physical:* Mobility and speech were modeled as physical user abilities that influence interaction with UIs. Regarding mobility, concepts such as brain motor area, precision, strength, and tactile sensation [21] were included. In the case of speech, context information about sound articulation and voice features were added.
3) *Sensory:* Sight, hearing, and human touch senses were included because they are directly related to interaction abilities [22].
4) *Affective:* Both the transmission and the interpretation of emotions are a key part of human communication. For instance, people who have affective disorders, such as Asperger syndrome, usually have communication problems (both verbal and nonverbal) [23].

The affective interaction ability is only considered in the representation of the user model, not in the application of the adaptations. Ontologies can represent knowledge of real-world domains using a large number of concepts, but it is difficult and often not necessary to use the whole model for all users or scenarios (for instance, in some cases, using the description of cognitive processes in detail will not be necessary).

The "level" property (bottom right part of Fig. 3) indicates the level of each ability within a specific scale. The scales for this property are similar to the measures of the ontology proposed by [24]: High (H), Medium (M), Low (L), and Not Applicable (N/A). In some domains, it is sufficient to use general categories and not complex metrics because the adaptation possibilities of a system are limited; whereas, in other domains (for instance,

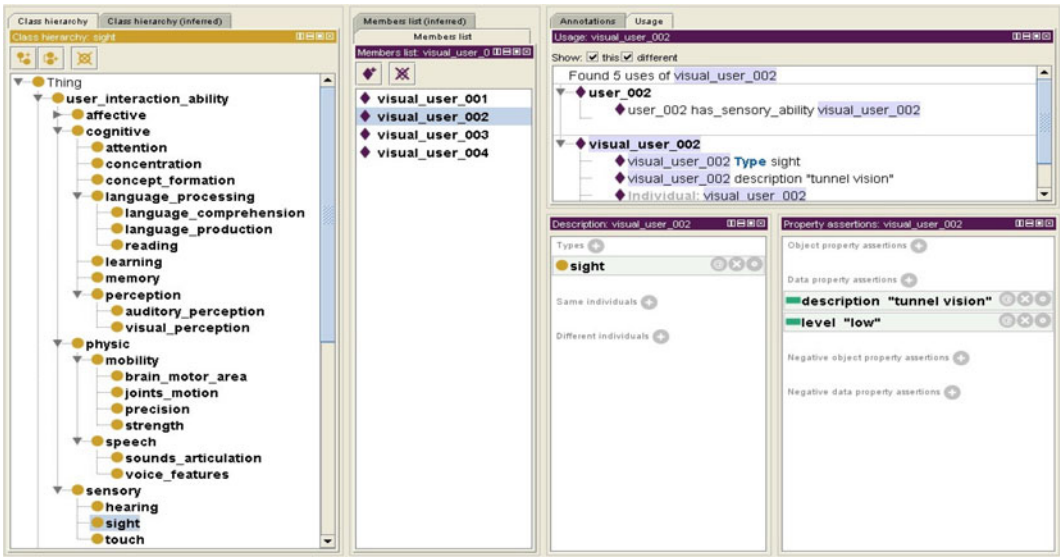

Fig. 3. Screenshot of the Protégé browser of Egonto showing the user-model class hierarchy for an individual with tunnel vision.

vision or hearing impairments), specific scales and metrics could be necessary. To this end, a field named "description" is included. A specific metric can be added to this field (by means of textual terms) to distinguish different abilities and better personalize them. For instance:

1) In addition to indicating that the value of the "level" property is "L" in the sight sensory-related ability, the "tunnel vision" description is included (see the representation of this case in Fig. 3).
2) The "From birth" description is added in a user profile with a "not applicable" level in the hearing sensory-related ability.
3) More specific information is included for visual perception, such as "problems in color perception" or "problems in distance perception" [24].

Users' abilities can also be combined to establish a more specific user profile. Consequently, it is possible to model users with combined disabilities. The combination of user abilities is particularly useful for modeling characteristics of elderly people because they usually experience more than one type of restriction related to hearing, sight, motor skills, memory, and reasoning [21]. People who have a combination of impairments need specific adaptations. For instance, separating the elements in the interface facilitates their selection for a user who has mobility-related precision restrictions. In the case of a user with tunnel vision, it may be more appropriate to concentrate the objects in the center of the interface. However, a person who has both movement precision problems and tunnel vision would need a different adaptation. Multidisability cases are addressed by means of reasoning rules included in the ontology. The use of reasoning rules is explained in Section III-C4.

2) Device Model: Device characteristics are also modeled in Egonto [see Fig. 4(a)] because required adaptations may be different depending on device-related issues. For instance, the adaptations required by a smartphone user may differ from the adaptations required by a tablet user. In addition,

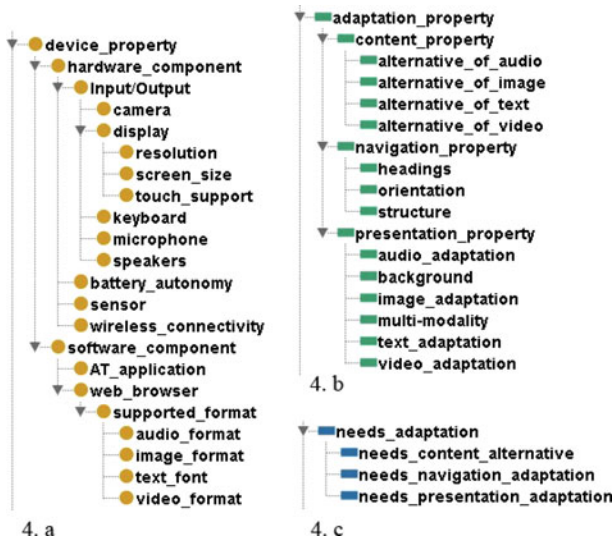

Fig. 4. Screenshot of the Protégé browser of Egonto. (a) Device model class hierarchy. (b) Object properties for linking the user concept with the adaptation concept. (c) Data properties to indicate the scope of the adaptations.

accessibility is strongly linked to the user's preferred modality. Therefore, it is advisable to model the type of modalities that can be used with a specific device.

There are two types of general ontology classes in the device model [see Fig. 4(a)].

1) *Hardware components:* hardware-related features such as input/output devices; peripheral features (such as display resolution); interaction modes allowed (e.g., touch screen); battery autonomy; and wireless connectivity.
2) *Software components:* information about the applications installed on the device. This includes Web browser features, such as whether JavaScript is supported, which versions of markup languages can be interpreted, which formats are supported, and which assistive technology

applications are installed on the device (e.g., screen reader, face detection, or voice recognition).

3) Adaptation Model: To represent the adaptations, *Egonto* follows an adaptation taxonomy based on [25]. This taxonomy considers content, navigation, and presentation adaptations. *Egonto* includes these concepts to match each user with the required adaptations. Fig. 4(b) shows the properties for linking a user with the adaptations set, and Fig. 4(c) depicts the properties that the ontology uses for indicating the scope of the adaptations.

Egoki performs content adaptations to select an alternative resource when the user does not have the ability to use a specific media type for a particular resource. For instance, simplified text can be provided as an alternative to the usual text media type for people with reading difficulties. Four generic media types are described, and some alternative resources are indicated as a content adaptation for each of them. Some examples of alternative resources for each generic media type are listed below.

1) Alternative for video media type: captions.
2) Alternative for audio media type: transcriptions.
3) Alternative for text media type: simplified text or pictograms.
4) Alternative for image media type: pictograms or alternative text.

The interface presentation is adapted by varying parameters such as the font color and size, the background color, the size of images, or selecting high-contrast resources.

Regarding navigation adaptations, in some cases, the UI should be divided to support comprehension of the current task. When the task requires several steps, a possible navigation adaptation is to provide the user with a task sequence describing each step in detail.

4) Reasoning Rules: Some generic rules were included to cover the needs of specific groups of people with special needs (such as vision, hearing, cognitive, and motor impairments). These rules allow an initial profile to be assigned to each user and were defined by means of ontology concepts: sight, hearing, and touch for the sensory capability; cognitive capability; and speech and motor skills for the physical ability. In addition, more specific rules have also been included to allow more detailed adaptation. Specific properties also allow multiple impairments to be taken into account.

If a given user requires a different profile or more specific adaptations, this information is collected and included in Egonto. This way, the ontology can learn the missing feature and can include a new rule covering a profile with these characteristics.

The above-mentioned reasoning rules were added into Egonto using the Semantic Web Rule Language (SWRL) [26], and they are applied from the KB module using the Pellet reasoner. Fig. 5 shows a generic rule for people with a high level of sensory and physical abilities and a medium level of cognitive abilities. The result of applying the reasoning rule is a set of interface adaptations to be applied by the Egoki system. To this end, a study of the needs of people with different abilities was performed. Based on the analysis of required adaptations, we have constructed and stored different SWRL rules in the ontology. Having these rules

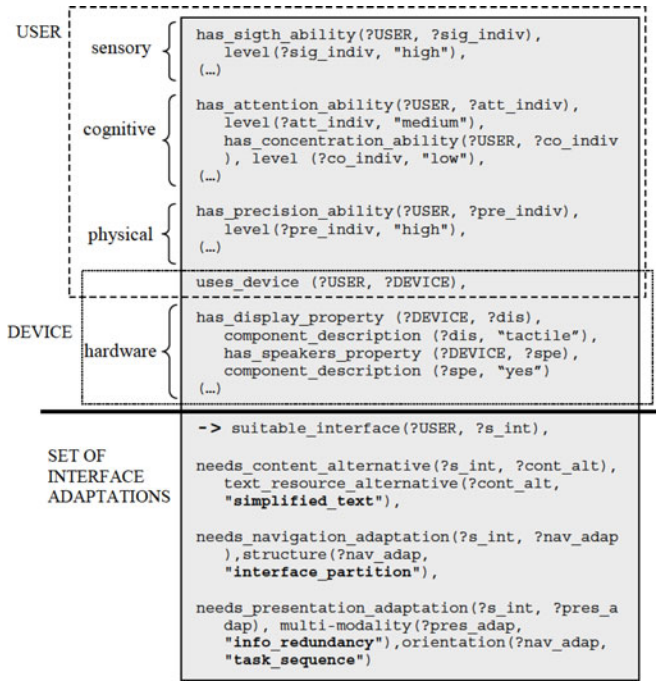

Fig. 5. Screenshot of the Protégé Rule view showing a reasoning rule that assigns a suitable adaptation set for specific communicative abilities and device access features.

in the ontology, Egoki will apply the corresponding rule to the specific features of a certain user. In this example, simplified text is required as an alternative resource to the text content because the user's reading level is low. This adaptation facilitates comprehension of the UI functionalities as well as the content. Regarding navigation, the UI is divided into several steps to help the user perform the task. With respect to the presentation, audio, simplified text, and image are added for each resource. Providing multiple media for resources supports understanding and abstraction of concepts. In addition, the audio output provides additional verbal feedback to the user.

5) User Interface Model: As previously stated, Egoki requires a logical description of the UI, describing the functionality of the ubiquitous services, to generate a final user-tailored interface. A number of user-interface description languages such as UsiXML [3] and MARIA [4] were analyzed. UIML was finally adopted, as it provides the necessary semantics to easily include alternative types of resources. In addition, using UIML is easier for service providers than UsiXML and MARIA. UIML is an XML-based language that provides elements and attributes for defining the structure, content, and behavior of the UI (see Fig. 6). However, this language does not provide direct mechanisms for defining some necessary elements such as dependences on functionalities and interaction elements. It also lacks a way to set priorities for the different alternatives of types of resources. It is flexible enough to be extended by means of a specific vocabulary. Therefore, we defined an extension of UIML to enhance its expressiveness.

Next, the new features included in the UIML vocabulary are introduced. We will use a ubiquitous service called "lunch menu

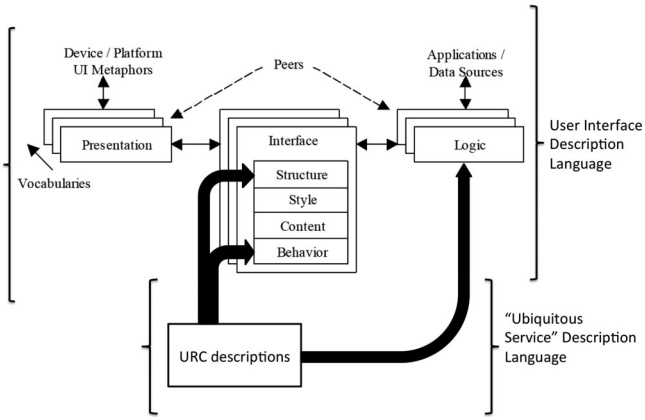

Fig. 6. UIML metainterface model modified from [27].

```
<part id="FirstCourseSection" class="section">
    …
</part>
<part id="SecondCourseSection" class="section">
    …
</part>
<part id="DessertSection" class="section">
    …
    <part id="DessertFunctionality" class="functionality">
        <part id="Dessert" class="select">
            <content id="priority">
                <constant id="text"  value="1"/>
                <constant id="image" value="2"/>
                <constant id="audio" value="3"/>
                <constant id="video" value="4"/>
            </content>
            <variable name="inputDessert"/>
            <part id="Dessert_a" class="item" source="Dessert_a"/>
            <part id="Dessert_b" class="item" source="Dessert_b"/>
            <part id="Dessert_c" class="item" source="Dessert_c"/>
        </part>
        …
    </part>
    …
</part>
```

Fig. 7. Segment of code showing the definition of different types of resources for the "selecting a dessert" functionality.

selection" to illustrate them. For this purpose, excerpts of the code for this service are presented in Figs. 7–9.

The UIML vocabulary extensions are as follows.

1) The *class* attribute of the *part* element in UIML is used for defining the type of interaction element. Possible values for this attribute are *section*, *select*, *item*, *button*, *outputData*, *boolean*, *inputData*, and *functionality*. The first value (*section*) is used to group related interaction elements so that they can be placed close to each other in the final UI. The *functionality* type is used to define parts of the interface as functionalities that are directly operated by users. The other elements identify the type of interaction elements included in the content. Fig. 7 shows the description of a ubiquitous service called "lunch menu selection." In this example, the service is composed of three *section* type elements specified by means of the class attribute. In the last *section* type element, a *functionality* type element is provided with a single *select* type interaction element with three options.

```
<content id="text" >
  <constant  id="ref_text:Dessert_a"
             value="Fresh fruit apple"/>
      <constant id="ref_simplified_text:Dessert_a"
             value="Apple"/>
      …
</content>

<content id="image" >
      <constant id="ref_image:Dessert_a"
             value="resources\menu\image\dessert_a.png" />
      …
</content>

<content id="high_contrast_image" >
      <constant id="ref_high_contrast_image:Dessert_a"
             value="resources\menu\image\dessert_a_HC.png" />
      …
</content>

<content id="audio" >
      <constant id="ref_audio:Dessert_a"
             value="resources\menu\audio\dessert_a.mp3" />
      …
</content>
```

Fig. 8. Segment of code showing the definition of different types of resources for the functionality of "selecting a dessert."

2) Values to specify priorities for the different media types for the resources have been added to the vocabulary to allow service providers to define them. These values permit the Egoki system to select appropriate media types for resources according to user, device, and service features. Priorities must be set for each interaction element represented in the UIML. They are specified by the element *content* and value *priority* for the *id* attribute. In Fig. 7, the service provider has specified priorities for four types of media resources (text-priority 1, image-priority 2, audio-priority 3, and video-priority 4).
3) Different values have been defined for the id attribute of the *constant* element in UIML for providing information about interaction resources: text, simplified text, video, image, high contrast image, audio, simplified audio, text transcription, and signed video. Fig. 8 shows examples of different types of interaction resources.
4) The data exchange between the Egoki system and the ubiquitous services is modeled in the *behavior* section of UIML. The behavior of UIs is described through *rule* elements. Some extensions have been made to the vocabulary to accommodate event-driven code generation. *Init, change*, and *select* are the events used for UI elements. Fig. 9 shows the *behavior* section of the *select type* element for selecting the dessert in the "lunch menu selection" service.

As previously indicated, service providers are responsible for handling the UIML description as well as providing different types of resources for interaction elements. This can require the use of a wizard tool, similar to the one presented by Miñón *et al.* [28], with a twofold objective: to allow service providers to create valid UIML documents with less effort; and to assist in the selection of appropriate resources for the abstract UI.

```
<behaviour>
 <rule id="rule_Dessert">

  <condition>
   <event part-name="Dessert" class="change"/>
  </condition>

  <action>
   <call component-id="GuresareMeal" methodid="setDessert">
      <param name="setDessert_param1">
       <property part-name="Dessert" name="inputDessert"/>
      </param>
   </call>
  </action>

 </rule>
</behaviour>
```

Fig. 9. Segment of code showing the definition of different types of resources for the functionality of "selecting a dessert."

The middleware layer (URC/UCH) includes a type of UI description language that, similarly to UIML, addresses device independence and multimodality of the UI [29]. However, we found that UIML is better suited to our requirements for the structure of the UI, which plays a key role in the UI generation process. Therefore, we rely on the URC for the description of the ubiquitous services and on the UIML for the UI description necessary to operate the ubiquitous service, integrating the URC description language in the UIML (see top of Fig. 6). The UIML files in the current version of Egoki include information to call URC/UCH to access the ubiquitous services. This approach allows replacement of the middleware layer (e.g., following the architecture of multimodal interfaces proposed by the World Wide Consortium [30]) with only minor impact on the Egoki system.

D. User-Tailored Interface Generation Process

In summary, the automated generation of user-tailored interfaces is based on the transformation of a logical specification of the UI described in UIML into a final functional UI using the information specified in the models. The process that Egoki performs to generate automated user-tailored interfaces consists of three main phases: querying models, selecting resources and adaptations, and constructing the final UI (see Fig. 2).

1) Model Queries: The RS module parses the UIML document and retrieves the information on functionalities, including the media resource types for interaction elements and priorities. The RS module gets each part element of the UIML that contains a "functionality" value in the class attribute, as well as the child elements inside it.

2) Media Resource and Adaptation Selection: The RS module queries the KB to obtain a specific type(s) of interaction resource to represent each functionality according to user and device features. See Algorithm I, which selects the appropriate resource(s).

As output from this phase, the RS module provides the AE module with the list of specific resources as well as possible adaptations to better accommodate the user's needs. The AE later uses this information to generate the final UI.

ALGORITHM I
RS MODULE ALGORITHM FOR SELECTING APPROPRIATE RESOURCE(S) FOR A GIVEN USER

```
For each functionality of the service
1-Select the media type with the highest priority
2-Query the ontology about the suitability of this
     media type for the specific user and his/her
     device or the need for an alternative resource
If the media type is not appropriate and an
     alternative is not provided:
select the media type with the next highest priority
     and go to step 2 again
Else select the resource associated with this media
     type and go to step 3
3 - Query the ontology about the adaptations
     required for that media type according to the
     user's ability levels and access device
     features.
```

```
<audio id="ref_audio:Dessert_a"
    preload="auto">

 <source type="audio/mpeg"
         src="resources\menu\audio\dessert_a.mp3"/>

</audio>

<input id="Dessert_a" type="radio" name="Dessert"
       value="Dessert_a"/>

 <label for=" Dessert _a">
  <img alt="Apple" src="resources\menu\image\dessert_a.png"/ >
  <span>An apple</span>
 </label>
</input>
```

Fig. 10. Excerpt of the HTML 5 code generated by Egoki.

3) Final User Interface Generation: In this last step, the AE module generates a functional final UI tailored to the needs of the user and device. The AE module applies several XSL transformations and CSS rules. There are three main types of transformations for different generation purposes: content, behavior, and presentation.

a) Content: XSLT rules are used to transform logical UI descriptions specified in UIML into specific interaction elements in the HTML language. HTML was selected for these UIs because they interact with ubiquitous services operated by mobile devices and HTML is the most common language for these devices. Fig. 10 shows an excerpt of the HTML code generated for the "lunch menu selection" service, where the interaction element is composed of audio, text, and image resources.

b) Behavior: UCH middleware provides specific URLs in order to *set* and *get* the state of the ubiquitous services used for defining the behavior. This information is contained in the *Peers* section of the UIML document (see Fig. 6). The *Behavior* section identifies the functions triggered by each interaction element and the supported events.

This way, specific XSLT rules were created to provide the final UIs with behavior.

Step 1 (File generation): A javaScript file is generated for each ubiquitous service.

Step 2 (Event registration): The events supported by each interaction element are bounded to their correspondent HTML elements. This allows generating non-intrusive javaScript code.

Step 3 (Function generation): A javaScript function is generated for each rule defined in the *Behavior* section. These functions *get* and *set* the states of the ubiquitous services. To this end, the URLs provided by the middleware are included in the function, which uses Ajax technology to communicate with the services.

c) Presentation: CSS rules are used to implement some style-related adaptations for accommodating user needs, for example, setting foreground and background colors, font styles and sizes, and space between UI elements.

By executing content, behavior, and presentation transformations, the AE module is able to generate a Web UI that is tailored to the user's needs. In addition, the UI allows the user to use the requested ubiquitous service. The markup language used in the final UI will mainly depend on the requirements of the access device.

E. Implementation

The Egoki system was developed using the Eclipse version for Java EE developers [31]. Egoki is a Java Web application compliant with JDK 1.6. Additional libraries include the following:

1) Pellet [19] and OWL API [32] for accessing ontology and inferring information by applying SWRL rules;
2) Apache Xerces [33] for parsing, manipulating and creating different XML documents;
3) Apache Xalan [34], an XSLT processor for transforming XML documents to XML, HTML, or text;

The Egonto ontology was developed using OWL [35], and the rules included in Egonto are based on SWRL [26].

As mentioned, the final UIs are web interfaces based on HTML, and their behaviors were generated using jQuery [36] to manage UI events and Webclient Javascript library [37] to provide necessary interaction with the ubiquitous service.

IV. Case Study

To evaluate the operability and accessibility of the automatically generated UIs, two services were implemented using the Egoki system. These services support people with disabilities in performing daily activities. The first (Service 1) was a *lunch menu selection service*, and the second (Service 2) was a *local bus information service*. Both were defined using UIML specifications, and a set of multimodal resources were created for each interaction element.

The lunch menu selection service allows users to select a first course, a second course, and a dessert. This service was based on the paper-based menu currently used in a local shelter workshop for people with cognitive disabilities. The local bus information service allowed searching for urban bus routes and schedules. The service is based on a nonaccessible online service provided by the local public transport company.

Both UIML specifications and multimodal resources were included in the Egoki system, and two scenarios were simulated to evaluate the adaptation process of the system. The first scenario (Scenario 1) was designed for blind users, while the second (Scenario 2) was designed for people with cognitive impairments.

TABLE I
Adaptation Techniques Applied in Scenario 1

	Scope	Adaptation
Content	Image media type alternative	"text"
	Audio media type alternative	"text"
Navigation	Headings	"heading inclusion"
	Structure	"interface partition"

Fig. 11. UI for selecting a dessert, created by the Egoki system in Scenario 1 [Translation to English: "Menú" is "Menu," "Postre" is "Dessert," "Selecciona el postre" is "Select a dessert," "Flan" is "Crème Caramel"; "Menú de Navegación" is "Navigation Menu"; "Anterior" is "Back" and "Siguiente" is Next.].

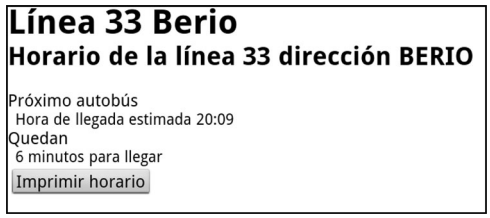

Fig. 12. UI with the bus information obtained in Scenario 1 [Translation to English: "Línea 33 Berio" is "Line 33 Berio" "Horario de la línea 33 dirección Berio" is "Schedule of bus line 33 bound to Berio" "Próximo autobus. Hora de llegada estimada 20:09." is "Estimated arrival time of next bus 20:09"; "Quedan 6 minutos para llegar" is "6 min to arrive"; "Imprimir horario" is "Print Schedule"].

The adaptation techniques applied for both scenarios differed. In Scenario 1, "text" resources were included for each element of the interface because text is the most suitable media type for screen readers. Headings were also added to the interface (using the "heading inclusion" technique). Instead of presenting large amounts of information in just one UI, the information was split into a number of simpler UIs ("interface partition"). Table I shows the adaptation techniques applied in Scenario 1.

Fig. 11 shows a screenshot corresponding to Service 1 generated for Scenario 1. The UI only includes textual resources. Due to the "interface partition" adaptation, two new link elements were generated: "*Anterior*" (Spanish for "Back") and "*Siguiente*" (Spanish for "Next"), to allow for navigation between the different steps.

Fig. 12 shows the UI corresponding to Service 2 created by Egoki for Scenario 1. Figs. 13 and 14 show the UIs for Services 1 and 2 in Scenario 2.

Table II shows the adaptation techniques applied in Scenario 2. In this case, "simplified text" resources are used for

TABLE II
ADAPTATION TECHNIQUES APPLIED FOR SCENARIO 2

	Scope	Adaptation
Content	Image media type alternative	"info redundancy" (image, audio, simplified text)
	Text media type alternative	"info redundancy" (image, audio, simplified text)
	Audio media type alternative	"info redundancy" (image, audio, simplified text)
Navigation	Orientation	"task sequence"
	Structure	"interface partition"

Fig. 13. UI for selecting a dessert, created by the Egoki system in Scenario 2 [Translation to English: "Primer plato; Segundo Plato; Postre Confirmar" is "First Course; Main Course; Dessert; Confirm"; "Postre" is "Dessert"; "Escoge lo que quieres comer de postre" is "Select a dessert"; "Flan" is "Crème Caramel"; "Yogur" is "Yogurt"; "Manzana" is "Apple"; "Anterior" is "Back" and "Siguiente" is "Next"].

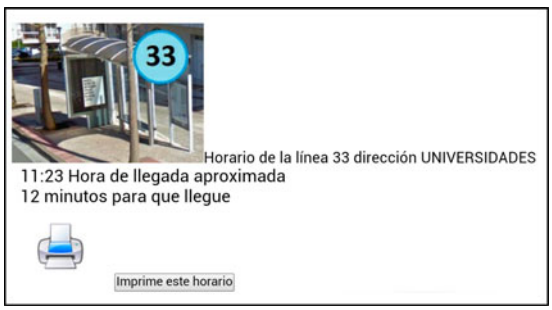

Fig. 14. UI for local bus information service created by the Egoki system in Scenario 2. [Translation to English: "Horario de la línea 33 dirección UNIVERSIDADES" is "Schedule of bus line 33 bound to UNIVERSITIES"; "11:23: Hora de llegada aproximada" is "Approximated arrival time 11:23"; "12 minutos para que llegue" is "12 min to arrive"; "Imprime este horario" is "Print this Schedule"].

representing each element of the interface instead of using ordinary textual resources. Two other types of resources were added: audio and image resources. The adaptation called "info redundancy" was specifically implemented for users with cognitive restrictions who need redundant interaction modes to reinforce their understanding. Another adaptation was the inclusion of a special component called "task sequence." This informative element shows the whole sequence of steps in a UI, where the current step appears highlighted. Finally, the previously mentioned "interface partition" adaptation was also applied to simplify the interface. This adaptation was necessary not only to simplify the interface but also to avoid accessibility issues with regard to dynamic content, for instance, when the requested information from a service has to be displayed on the UI.

A. Evaluation

We performed two different types of evaluation of the automatically generated service interfaces: expert-based evaluation and user evaluation. Expert-based evaluations were carried out to identify accessibility barriers for people with disabilities. User evaluation sessions were performed with the objective of evaluating the operability and accessibility of the automatically generated UIs.

1) Expert-Based Evaluation: Expert evaluation was based on the barrier walkthrough (BW) accessibility inspection method [38]. Six accessibility experts performed this evaluation. The interfaces for both services were evaluated for both scenarios; three experts evaluated Services 1 and 2 for blind users, and the other three evaluated both services for users with cognitive disabilities. Two interfaces were evaluated for each service: an option selection interface and the confirmation/print interface.

Evaluators were provided with the documentation required to perform the BW method, and each evaluator independently checked all barriers detected in the interfaces of both services. Then, they met and merged the detected barriers into a single list by assigning a single severity score to each barrier. Table III summarizes the results.

Results from the BW showed that all the barriers found were minor for Scenario 1. Analyzing the barriers identified for Scenario 1, we realized that all of them were side effects of the application of the "interface-partition" adaptation. That particular adaptation was required for two reasons: first, to simplify the interaction (as Service 1 involves several steps to select a specific menu), and second, to address the dynamic content generated by the selection of dishes. As the adaptation offers advantages to the interaction with a screen reader, our hypothesis was that the adaptation would be favorable and would benefit users even if it introduced potential minor barriers.

With regard to Scenario 2, no critical barriers were identified. We realized that the barrier "no page headings" was present because the adaptation for including headings was not applied. This knowledge would be added to the Egonto KB if the results of the user evaluation showed that it was required. The barriers of "generic links" and "ambiguous links" (in Service 1) were due to the application of "interface partition" adaptation. The barriers of "complex text" and "rich images lacking

TABLE III
ACCESSIBILITY BARRIERS DETECTED IN BOTH SCENARIOS FOR SERVICE 1 AND SERVICE 2 AND THEIR SEVERITY SCORE

Scenario 1: Blind users		
Severity	Barriers in Service 1	Barriers in Service 2
Minor	Page without titles Generic links Non-separated links	-
Significant	-	-
Critical	-	-
Scenario 2: Users with cognitive disabilities		
Severity	Barriers in Service 1	Barriers in Service 2
Minor	Complex text No page headings Missing icons	Rich images lacking equivalent text Acronyms and abbreviations without expansions Complex text
Significant	Rich images lacking equivalent text Generic links Ambiguous links	No page headings Missing icons
Critical	-	-

equivalent text" are related to the provided resources, not to the Egoki system itself. The same applies to the "missing icons" barrier, which is related to the information requested from the service. As the information comes in real time, Egoki displays the information as provided by the service. If the service does not provide the information in different modalities, Egoki cannot create alternative resources dynamically. With regard to the "acronyms and abbreviations without expansions" barrier, because we found no acronyms or abbreviations on the generated UIs, we did not consider that a barrier.

2) User Evaluation: Eleven participants took part: two legally blind participants (a 29-year-old female and a 39-year-old male) and nine participants with cognitive disability (four females and five males within an age range of 27–43 years).

With regard to the apparatus, blind participants were required to bring their personal laptop. Thus, the devices were already configured according to their preferences and needs. Both blind participants used the same model and version of the screen reader (JAWS v.12) but with different Web browsers (Mozilla Firefox 4 and Internet Explorer 9). The participants with cognitive disabilities all used the same device, an off-the-shelf tablet with the Android 4.0.3 operative system and the Google Chrome Web browser. These touch-screen devices had been recommended by their tutors.

We used the previously described two services as stimuli: a lunch menu selection service (Service 1) and a local bus information service (Service 2).

The user evaluation sessions were conducted on two different days and in two different locations. On the first day, we conducted the sessions with the blind participants in a lab at the Computer Science School at the University of the Basque Country. The sessions with the participants with cognitive disabilities were carried out on a different day and in a room at the GUREAK shelter workshop. We followed similar procedures in both cases. Sessions were conducted one by one (one participant at a time) and in Spanish, the first language of all participants. Each session was started by providing the participant with the information about the study. After that, they were asked for their consent to record the session. They were asked some questions to collect demographic data (e.g., age, experience with computers, and experience with touch screens or screen readers). In the case of participants with cognitive disabilities who had never used a touch screen device, some explanations were given and they were allowed to try the tablet before starting the evaluation.

Following a "within subject" approach, all participants had to use both services and complete two tasks in each; a fixed task (Task 1) and a personal task (Task 2). The fixed task was the same for all participants: in the case of the Service 1, to select a menu composed of three particular dishes; and for Service 2, to obtain information on a given bus line. The personal tasks were the same as the fixed tasks, but they could choose different options; for instance, the dishes from the menu in Service 1 and the bus line in Service 2. The Service order was counterbalanced to avoid order effects. Each evaluation started with the first task of the corresponding service. When the first task had been completed, the participant started on the second task. When both tasks for a service had been completed, the participant was interviewed. After that, the participant was allowed to start on the other service.

In each session, the following data were collected: 1) Logs—Every user interaction with the ubiquitous services was monitored and stored in log files. 2) Video recordings—Two cameras were used to record user interactions from different perspectives. One camera was focused on the screen of the device and the other was focused on the participants. 3) Experts' notes—Interaction-specific aspects that drew the attention of the investigator were noted. 4) Information about participants' satisfaction, their opinions, and problems identified was gathered through short postinteraction interviews.

B. Results

The participants successfully completed Tasks 1 and 2 in both scenarios. Each participant applied different strategies when completing the tasks. Participants with cognitive disabilities required a varying amount of support from their

TABLE IV
TASK COMPLETION TIMES FOR SCENARIO 1 (IN SECONDS)

	Service 1		Service 2	
Participant	Task 1	Task 2	Task 1	Task 2
P01	93	62	53	45
P02	230	565	188	65

TABLE V
TASK COMPLETION TIMES FOR SCENARIO 2 (IN SECONDS)

	Service 1		Service 2	
Participant	Task 1	Task 2	Task 1	Task 2
P03	370	375	360	85
P04	325	265	555	160
P05	120	135	110	30
P06	70	60	105	40
P07	260	32	270	75
P08	400	165	175	115
P09	610	240	315	40
P10	140	185	310	75
P11	245	140	60	95

tutors depending on their cognitive abilities and the tasks they had to complete.

Tables IV and V show task completion times for blind participants (Scenario 1) and participants with cognitive disabilities (Scenario 2), respectively.

Most participants needed more time to finish Task 1 than to finish Task 2 in both services (seven participants in Service 1 and ten participants in Service 2).

Regarding Service 1, we determined that, for two of the participants (P03 and P05), the completion time was slightly longer for Task 2 than for Task 1 (5 and 15 s, respectively). Participant P10 took longer to complete the second task (45 s). We detected some difficulties when interacting with the tablet; for example, tapping on the screen too gently or too hard. Participant P02 voluntarily performed a sort of think-aloud protocol (during Task 2, he was narrating his interaction with different interface elements), and it took him longer to finish the task.

Regarding Service 2, only P11 required more time for Task 2 (35 s). By analyzing the recordings, we realized that she was unsure about which bus to choose.

C. Discussion

From the results obtained, we concluded that the UIs generated by Egoki for both samples (blind participants and participants with cognitive disabilities) are operable because they allow accessing all the services' functionalities. Regarding their usability, participants commented in the postinteraction interview that the UIs were very easy to use. Nevertheless, the analysis of video recordings revealed a number of issues to be considered.

1) Issues With the Egoki Models: A number of issues in the models used to feed Egoki for the creation of the interfaces were detected. In Scenario 1, blind participants found adequate the provision of textual resources. The "heading inclusion" adaptation technique was also appreciated, as it facilitated the interaction with screen readers. However, they did not like the "interface partition" adaptation technique. Specifically, participant P02 suggested the creation of only one UI for selecting all options in Service 1, as this would increase the efficiency of interfaces. He also missed access keys, which in his opinion would have improved his performance. This means that the model used by the Egoki to produce UIs for blind people has to be adjusted.

The "info redundancy" adaptation technique played a key role for participants in Scenario 2. Participants initially paid attention only to the pictures and icons. However, the associated audio resources seemed to be the most helpful elements. Most of them ignored textual resources. The "interface partition" adaptation technique was essential for them. Due to their difficulties in maintaining attention, it was important to keep interfaces as simple and clear as possible. On the other hand, analysis of the video recordings revealed that the "task sequence" adaptation technique was not beneficial; some participants simply did not pay any attention to it, while others found it confusing. Although it was intended to inform them about their progression in task completion, five of nine participants tried to press it, taking it for a button. Therefore, this adaptation technique needs to be analyzed and redesigned to be valuable for this type of user.

2) Issues With the Final User Interface Generation: The analysis of participant interactions also revealed some issues related to the "presentation" adaptation rules applied by the Egoki system. We observed that it was difficult for some participants in Scenario 2 to find and press elements such as buttons. The problem seems to be related to their position in the UI. In the postinteraction interview, some participants suggested replacing buttons with suitable icons and associated audio resources. Several participants also commented that they would prefer bigger buttons.

Similar issues can also be caused by a lack of quality or variety of resources developed by the service provider. For instance, some buttons contained too long texts that hindered reading, especially for participants with low reading skills.

V. CONCLUSION

The development of supportive ubiquitous services requires that the UI downloaded to the user's mobile device be accessible. In this paper, we presented the Egoki system, which is able to automatically generate model-based UIs that meet a user's requirements.

The Egoki system created UIs for two different ubiquitous services that were evaluated by accessibility experts. In addition, a proof-of-concept prototype evaluation was carried out with blind participants and participants with cognitive disabilities. The results revealed that UIs generated by Egoki were accessible and operable. All participants were able to complete the proposed tasks. Most participants spent less time performing the assigned second task than they did performing the first one. This suggests that participants gained experience through the process. Nevertheless, this point must be confirmed with further analysis.

Several minor issues were detected by expert and user evaluations. A number of issues are related to the models designed for Egoki and others to the final UI generation process. These issues must be considered in developing subsequent versions of the Egoki system.

ACKNOWLEDGMENT

The authors would like to thank the people involved in the evaluation of the Egoki system, as well as A. Heredia Saenz from GITEK/GUREAK for providing support with the scenarios.

REFERENCES

[1] *INREDIS*. (2010). [Online]. Available: http://www.inredis.es/default.aspx
[2] R. Miñón, J. Abascal, A. Aizpurua, I. Cearreta, B. Gamecho, and N. Garay, "Model-based accessible user interface generation in ubiquitous environments," in *Proc. INTERACT*, 2011, vol. 4, pp. 572–575.
[3] Q. Limbourg, J. Vanderdonckt, B. Michotte, L. Bouillon, and V. López-Jaquero, "USIXML: A language supporting multi-path development of user interfaces," in *Proc. Eng. Human Comput. Interaction Interactive Syst.*, 2005, pp. 200–220.
[4] F. Paternò, C. Santoro, and L. D. Spano, "MARIA: A universal, declarative, multiple abstraction-level language for service-oriented applications in ubiquitous environments," *ACM Trans. Comput.-Human Interaction*, vol. 16, no. 4, pp. 19:1–19:30, 2009.
[5] M. Feld, G. Meixner, A. Mahr, M. Seissler, and B. Kalyanasundaram, "Generating a personalized UI for the Car: A User-adaptive rendering architecture," in *Proc. 21th Int. Conf. User Model., Adaptation Personalization*, 2013, pp. 344–346.
[6] H. Kaindl, E. Wach, A. Okoli, R. Popp, R. Hoch, W. Gaulke, and T. Hussein, "Semi-automatic generation of recommendation processes and their GUIs," in *Proc. 18th Int. Conf. Int. User Interfaces*, 2013, pp. 19–22.
[7] O. Corcho, A. Gómez-Pérez, A. López-Cima, V. López-García, and M. C. Suárez-Figueroa, "ODESeW. Automatic generation of knowledge portals for intranets and extranets," in *Proc. Int. Semantic Web Conf.*, 2003, pp. 802–817.
[8] R. Hervás and J. Bravo, "Towards the ubiquitous visualization: Adaptive user-interfaces based on the Semantic Web," *Interacting Comput.*, vol. 23, no. 1, pp. 40–56, 2011.
[9] *GUIDE (Gentle User Interfaces for Elderly People) European Project*. (2014). [Online]. Available: http://www.guide-project.eu/
[10] M. Peissner, D. Häbe, D. Janssen, and T. Sellner, "MyUI: Generating accessible user interfaces from multimodal design patterns," in *Proc. 4th ACM SIGCHI Symp. Eng. Interact. Comput. Syst.*, 2012, pp. 81–90.
[11] *AALuis: Ambient Assisted Living User Interfaces. European Project*. (2014). [Online]. Available: http://www.aaluis.eu/
[12] K. Z. Gajos, J. O. Wobbrock, and D. S. Weld, "Automatically generating personalized user interfaces with Supple," *Artif. Intell.*, vol. 174, pp. 910–950, 2010.
[13] S. Bongartz, Y. Jin, F. Paternò, J. Rett, C. Santoro, and L. D. Spano, "Adaptive user interfaces for smart environments with the support of model-based language," in *Proc. 3rd Int. Conf. AmI*, 2012, pp. 33–48.
[14] P. Brusilovsky and C. Peylo, "Adaptive and intelligent web-based educational systems," *Int. J. Artif. Intell. Education*, vol. 13, nos. 2–4, pp. 159–172, 2003.
[15] G. Calvary, J. Coutaz, L. Bouillon, M. Florins, Q. Limbourg, L. Marucci, F. Paternò, C. Santoro, N. Souchon, N. D. Thevenin, and J. Vanderdonckt, (2002). "The CAMELEON reference framework," *Deliverable 1.1, CAMELEON Project*. [Online]. Available: http://www.w3.org/2005/Incubator/model-based-ui/wiki/Cameleon reference framework
[16] G. Zimmermann and G. C. Vanderheiden, "The universal control hub: An open platform for remote user interfaces in the digital home," in *Proc. 12th Int. Conf. Human-Comput. Interaction*, 2007, vol. 2, pp. 1040–1049.
[17] B. LaPlant, S. Trewin, G. Zimmermann, and G. Vanderheiden, "The universal remote console: A universal access bus for pervasive computing," *IEEE Pervasive Comput.*, vol. 3, no. 1, pp. 76–80, Jan.–Mar. 2004.
[18] J. Abascal, A. Aizpurua, I. Cearreta, B. Gamecho, N. Garay-Vitoria, and R. Miñón, "Automatically generating tailored accessible user interfaces for ubiquitous services," in *Proc. 13th Int. ACM SIGACCESS Conf. Comput. Accessibility*, 2011, pp. 187–194.
[19] Pellet. (2013). [Online]. Available: http://clarkparsia.com/pellet
[20] Z-L. Lu and B. A. Dosher, "Cognitive psychology," *Scholarpedia*, vol. 2, no. 8, p. 2769, 2007.
[21] K. R. Masuwa-Morgan, "Introducing AccessOnto: Ontology for accessibility requirements specification," in *Proc. 1st Int. Workshop Ontol. Interactive Syst.*, 2008, pp. 33–38.
[22] D. Te'eni, J. Carey, and P. Zhang, *Human Computer Interaction: Developing Effective Organizational Information Systems*. Hoboken, NJ, USA: Wiley, 2007.
[23] R. J. Frey and L. Larry, Eds., *UXL Encyclopedia of Diseases and Disorders*. Farmington Hills, MI, USA: GALE CENGALE Learning, 2009.
[24] S. Karim and A. Min Tjoa, "Towards the use of ontologies for improving user interaction for people with special needs," in *Proc. 10th Int. Conf. Comput. Helping People Special Needs*, 2006, pp. 77–84.
[25] E. Knutov, P. De Bra, and M. Pechenizkiy, "AH 12 years later: A comprehensive survey of adaptive hypermedia methods and techniques," *New Rev. Hypermedia Multimedia*, vol. 15, no. 1, pp. 5–38, 2008.
[26] (2004). *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. [Online]. Available: http://www.w3.org/Submission/SWRL/
[27] C. Phanouriou, "UIML: A device-independent user interface markup," Doctoral dissertation. Dept. Comput. Sci., Virginia Polytech. Inst. State Univ., Blacksburg, VA, USA, 2000.
[28] R. Miñón, L. Moreno, and J. Abascal, "A graphical tool to create user interface models for ubiquitous interaction satisfying accessibility requirements," *Universal Access Inf. Soc.*, vol. 12, no. 4, pp. 427–439, 2013.
[29] S. Trewin, G. Zimmermann, and G. C. Vanderheiden, "Abstract representations as a basis for usable user interfaces," *Interacting Comput.*, vol. 16, no. 3, pp. 477–506, 2004.
[30] *Multimodal Architecture and Interfaces, W3C recommendation*. (2012, Oct. 25). [Online]. Available: http://www.w3.org/TR/mmi-arch/
[31] *Eclipse IDE for Java EE Developers*. (2014). [Online]. Available: http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/keplersr2
[32] *OWL API*. (2014). Available: http://owlapi.sourceforge.net
[33] *Apache Xerces*. (2012). Available: http://xerces.apache.org
[34] Apache Xalan. (2006). Available: http://xml.apache.org/xalan-j/
[35] *OWL*. (2013). Available: http://www.w3.org/2001/sw/wiki/OWL
[36] *jQuery*. (2014). Available: http://jquery.com
[37] *Webclient Javascript Library*. (2009). Available: http://myurc.org/tools/Webclient/
[38] G. Brajnik, "A comparative test of web accessibility evaluation methods," in *Proc. 10th Int. ACM SIGACCESS Conf. Comput. Accessibility*, 2008, pp. 113–120.

Authors' photographs and biographies not available at the time of publication.