Adaptive Multimodal Web User Interfaces for Smart Work Environments

Giuseppe Ghiani[a,*], Marco Manca[a], Fabio Paternò[a], Joerg Rett[b] and Atul Vaibhav[b]
[a]*CNR-ISTI, Via Moruzzi 1, 56124 Pisa, Italy*
[b]*SAP SE, Bleichstraße 8, 64283 Darmstadt, Germany*

Abstract. This article presents a solution for supporting adaptive user interfaces in work environments that require operators to move about in dynamic contexts for manipulating various physical objects. The solution architecture is built upon the use of logical languages for interactive applications integrated with context aware and adaptive features. The proposed architecture is able to adapt to select aspects of the context during run-time by communicating with a context server and applying the specified adaptation rules. In order to show the possibilities of the proposed solution, we report on its application in the development of an adaptive user interface for a warehouse picking system, and discuss the results of the associated tests.

Keywords: Adaptive service front-ends, Context-aware user interfaces, model-based user interface languages, Warehouse picking system

1. Introduction

Work environments are continuously becoming richer in sensors, objects and devices and it is important that users are efficient and perform well in such contexts, without having to spend too much time to understand how to accomplish their tasks. Multimodal adaptive interfaces can be useful for this, because they can provide the required information in the most suitable modalities by taking into account the current context of use. For this purpose, it is useful to consider various contextual aspects, including user-related ones (tasks to accomplish, personal preferences and knowledge, etc.), technological parameters (available interaction resources, connectivity support, etc.) as well as the environmental dimension (level of noise, light, etc.).

In order to show the possible application of our methods, tools and languages, we consider a specific application domain, i.e. warehouse picking, which is a part of a logistics process often found in retail and manufacturing industries. Warehouses store the goods and products incoming from suppliers until they are collected and shipped to the stores or customers. The process of picking items from a shelf, collecting them in some sort of container and bringing them to certain locations is usually conducted by one or more persons. This leaves a maximum degree of freedom for the enterprise to change or rearrange the environment when needed. The costs of maintenance and down-times are lower compared to a fully automated system based on e.g. conveyor belts. In addition, differently from automated systems, human beings are well able to react and adapt to unforeseen situations.

Since the beginning of warehouse picking, a number of studies have been carried out on how pickers can be supported in their task. Technological solutions have been provided, such as Voice-directed warehousing (VDW). VDW refers to the use of voice direction and speech recognition software in warehouses and distribution centres. In a voice directed warehouse, workers wear a headset connected to a small wearable computer, which tells the worker where to go and what to do using verbal prompts. Workers confirm their tasks by speaking pre-defined commands and reading confirmation codes printed on locations or products throughout the warehouse.

A significant drawback of VDW is that the information is volatile. Once information, such as the

[*]Corresponding author. E-mail: giuseppe.ghiani@isti.cnr.it.

amount of items to be picked has been given, the system might not be able to repeat it. If the picker forgets such information, its retrieval might become laborious. Thus, visual UIs have gained importance in the task of supporting the picker.

Generally speaking, a modality is a way to interact with a system by exploiting one specific human sense. The level of multimodality, i.e. how two or more modalities are combined, can vary. In order to describe the possible combinations of modalities some authors use the CARE (Complementarity, Assignment, Redundancy, Equivalence) properties [3]. We think that the choice of the most suitable combination should depend on the contexts of use, which are becoming more and more dynamic. In order to obtain the correspondent suitable multimodal user interfaces we have considered model-based languages [11]. They provide designers and developers with a general vocabulary for describing their solutions., and they can be refined in concrete terms for various interaction platforms in order to obtain implementations in a variety of languages, even for various combinations of interaction modalities. In the application domain we tackle, in which the combination of vocal and visual interaction has only been studied in a limited way so far, model-based approaches can thus provide useful support and deserve investigation. Our approach exploits extensible adaptation rules that can take into account the dynamic context and allow adding multimodality and tuning its level as well. For instance, in an environment that becomes noisy the interaction should rely mainly on the graphical part. In addition, also the user's preferences and the capabilities of the platform must be considered. In a smart environment, the picker should receive support for navigating through the location and picking the right items, and context information should be gathered and used to adapt the UI accordingly.

In this paper we present a solution consisting of an adaptive, context-sensitive UI which is based on an architecture for context-sensitive service front-ends. The solution is based on the use of model-based languages for interactive application descriptions in order to facilitate the possibility of deriving multimodal versions adapted to various contexts of use, particularly in terms of device interaction resources. Such languages have been considered for standardisation in W3C[1] because of their useful support in creating versions of interactive applications that adapt to different interactive devices. However, they have been mainly used in academic environments, with very few cases of application to real world case studies.

In the paper, after discussing related work, we provide some information about the requirements for the target warehouse application and the software architecture designed to support multimodal adaptive user interfaces for smart work environments. Next, we describe the adaptation rules, which are specified in terms of *event, condition, action* (ECA) rules, and the example application considered and how it varies depending on the context of use. We then show the corresponding prototype and report on a user test focusing on the adaptation rules considered. Lastly, we draw some conclusions and provide indications for future work.

2. Related Work

Previous work addressing warehouse picking [26] reports on a 12-participant within-subjects experiment, demonstrating the advantages of a head-mounted display based picking chart over traditional text-based pick lists, paper-based graphical pick charts, and mobile pick-by-voice systems. However, no multimodal solution was investigated in that study. A different study [1] aimed at comparing various design solutions for head-mounted displays with different levels of information. The order was not only presented on a graphical UI (GUI) but also supported by some kind of sensors installed in the environment. This system exhibited a kind of Ambient Intelligence by detecting the picker's action of reaching into a shelf and comparing the respective box and its containing item with the order from the backend system. Such systems based on head-mounted displays (HMDs) can be extended to support Augmented Reality, as shown in [20]. One limitation of these contributions is that they provide solutions that are implemented with ad hoc techniques and thus cannot be easily generalised to other similar applications.

A framework for context-aware adaptivity of user interfaces for the elderly (AALFI) has been recently proposed [2]. It aims to enable context dependent adaptation of UI applications for supporting elderly users during both days and nights. A major functionality in AALFI is to provide proper feedbacks/alerts, such as reminding the user to have a meal or close the house door. Our approach is different not only for the considered domain, i.e. a working environment, but also for the technological choices: AALFI is based on an agent platform, where each agent is de-

[1] http://www.w3.org/2011/mbui/

veloped in Java; our architectural modules instead rely on different technologies (e.g. Java, JavaScript, etc.) and may even exploit external modules (e.g. Google vocal support), which makes it possible to support Web-based applications.

Context dependent adaptation is also tackled by the DEMANES project[2], aiming to make systems capable of reacting to contextual events by adapting to the situation. The project pilots have addressed safe urban transport, airport management and smart homes, while we have considered a work environment.

As indicated in the work done by Oviatt and others [6, 18], multimodal interfaces are characterized by the fusion of different inputs involving both the system and the user. This is why human perception and processing of multimodal information have been extensively studied by cognitive psychologists. Dumas et al. [6] and Reeves et al. [18] have drawn up a set of guidelines for the design of multimodal interfaces. They did not explicitly focus on application domains involving human workers on the move, such as in smart warehouse environments. However, by leveraging the newest technologies, in our warehouse scenario we have designed and implemented novel input/output combinations, while still considering their guidelines. For example, flexibility in terms of the context of use, feedback, consistency and error prevention have been addressed.

Our approach to multimodal adaptive applications also meets most of the requirements cited by Sire and Chatty [23]. Indeed, we rely on the MARIA framework that provides an abstract, implementation independent language, which is considered to be modality independent. Starting from an abstract representation of a UI, it is possible to obtain one or more concrete representations (i.e. runtime versions). The MARIA framework also provides control structures such as conditions and loops, and event detection mechanisms, as indicated by such authors.

In the FAME approach [5], multimodal adaptation is managed by rules encoded in behavioural matrices. These are structures that describe how adaptation should take place and that are able to store usage information (i.e. past interactions). In our view, historical context data are separated from adaptation rules: the former are the result of context sensing; the latter express contextual events, conditions on event parameters, and updates to be done on the interface when such events occur. Another difference with respect to our work is that behavioural matrices are component-specific, i.e. a behavioural matrix has to be developed for each adaptable component, while our adaptation rules can involve an arbitrary number of heterogeneous UI components. Thus, in our platform, one adaptation rule may be applied even to several different applications.

As a result of its flexibility, our approach can manage multimodal adaptation of any interface component depending on any type of contextual information, differently from FAME, where each component is adapted according to its specific rules and usage.

Mobile applications often require highly-focused visual attention, which poses problems when it is inconvenient or distracting to continuously look at a screen (e.g., while walking). Aural interfaces support more eyes-free experiences, as users can primarily listen to the content and occasionally look at the device. However, designing aural information architectures remains a challenge. For example, recent studies [27] have highlighted that backward navigation is inefficient in the aural setting, as it forces users to listen to each previous page to retrieve the desired content. Thus, they have introduced topic and list-based back navigation strategies in order to enhance aural browsing and improve the navigation experience, reducing the perceived cognitive load. This shows the potential of multimodal interfaces in ubiquitous scenarios, but also the need for some specific design solutions, which depend on the interaction modalities exploited.

The problem of obtaining context-aware applications has been addressed through specific toolkits, e.g. [19], while we aim for a more modular solution able to separate the part dedicated to context monitoring from the application, and integrate them through the use of extensible adaptation rules. In this way it is not necessary to know all the needed context dependent adaptations in advance (i.e. at application development time), but it is possible to customize the adaptive, context dependent behaviour by providing adaptation rules that determine the changes according to dynamic events.

Various solutions for user interfaces adaptation in mobile scenarios have been proposed so far, but they tend to consider only graphical user interfaces, see for example Responsive Design [10], or W3Touch [12]. We aim to obtain solutions in which the adaptation consists in varying the level of multimodality: to what extent each modality is used and how modalities are integrated among themselves.

The problem of obtaining user interfaces that are able to be rendered on multiple types of platforms,

[2] EU DEMANES project: http://www.demanes.eu/

including multimodal and vocal ones, has been addressed in some previous work, but still needs more general, better engineered solutions. XFormsMM [7] is an attempt to extend XForms in order to derive both graphical and vocal interfaces. The idea is to specify the abstract controls with XForms elements and then use aural and visual CSS respectively for vocal and graphical rendering. However, aural CSS have limited possibilities in terms of vocal interaction and the solution proposed requires a specific ad-hoc environment. For this purpose we propose a more general solution which is able to derive implementations in various languages.

Obrenovic et al. [13] have investigated the use of conceptual models expressed in UML, in order to derive graphical, form-based interfaces for desktop, mobile or vocal devices. However, since UML is a software engineering standard, aimed at supporting the specification of the internal software application functionalities, it seems unsuitable to capture the specific characteristics of user interfaces. A different approach to multimodal user interface development has been proposed in [8], which aims to provide a workbench for prototyping UIs using off-the-shelf heterogeneous components. In that approach, model-based descriptions are not used and it is necessary to have an available set of previously defined components, which are able to communicate through low-level interfaces; thus making it possible for a graphical editor to easily compose them.

Sottet and others [21] have presented a set of general principles relevant for supporting model-based adaptation while in our case we present a software architecture supported by engineered tools that can be applied in real world applications. Octavia et al. [14] have considered the use of a model-based approach to facilitating adaptation in virtual environments, also using the event-condition-action paradigm. We provide a more general architecture for this purpose able to support adaptation involving various interaction modalities according to the current context of use.

To summarise, we can say that the few research proposals that have also considered multimodal interaction have not been able to obtain a suitable solution in terms of logical descriptions and corresponding software architectures, and provided limited support in terms of the generation of the corresponding multimodal user interface implementations. For example, in [22] transformations are specified through attributed graph grammars, whose semantics is formally defined but which have considerable performance limitations. Our solution can instead support adaptation at run-time according to the context of use. This is possible thanks to the performance of the underlying platform. In the following, we report on the platform architecture and on how the solution has been applied to a case study of industrial interest.
In addition, the application of multimodal interfaces in the industrial domain has been addressed in some previous work but still reveals issues concerning performance and usability. Raiyani and Kumar [17] discussed a multimodal concept in a warehouse scenario. The interface was based on a handheld-device capable of scanning barcodes, displaying visual information and receiving instructions and sending confirmations through a headset. However, their interface did not provide any information on how to reach the shelf and was perceived as too slow for warehouse workers. We are addressing these usability and performance issues by using context information on the users location based on Bluetooth beacons.

3. Requirements and Design Decisions for the Target Application

Our target application concerns a real retail warehouse. The application aims to support pickers in their task of collecting items from shelves by providing them with all the necessary information.

The business process has been introduced in [3] and was verified by interviews during an on-site visit at a central distribution centre. The interviewees were logistics staff working in warehouse picking, the head of the logistics centre and technical staff responsible for the IT infrastructure. As a result, the information needed to accomplish the relevant tasks in a solution able to suitably replace the existing pick-by voice system was identified: the shelf number of the next pick, and the number of items to be picked. Additional information, which according to those end-users and stakeholders can improve the efficiency, was indicated as well: current location in the warehouse, location of the next pick, and path to reach the next location.

This led to a first set of requirements:

- The number of the next shelf should be clearly indicated;
- The amount of items to be picked should be given when the picker is approaching the shelf;
- A map should indicate the current user and target positions and the path from the former to the latter;

- If the route is blocked an alternative pathway should be presented.

Additional requirements as stated by the end-users and stakeholders were:

- In addition to the existing vocal modality the visual modality should also be used;
- If the items to be picked are fragile, then the visual modality must not be used;
- If the environment is too noisy the vocal modality should not be used;
- The picker should be supported by the system in an unobtrusive way;
- The hands of the picker should be free to take hold of the items.

In advance we hypothesized that the main benefits of a HMD-based device (Smart Glasses) in contrast to a standard mobile device, e.g. smartphone, can be summarized in the following features:

- Head-up navigation: a head-mounted display allows users to follow the suggested route to the next pick without shifting attention from the environment to the device display;
- Hands-free interaction: since input/output does not require holding or touching the device, the hands are free for other activities, e.g. picking an item;
- Reality augmentation: virtual environment annotation conveys additional information but, differently from the handheld device display, does not imply visual and mechanical interference.

Furthermore we assumed that the advantages of Bluetooth beacons compared to other technologies supporting location-based services are:

- Indoor Localization: In contrast to GPS-based sensors, Bluetooth beacons can be distributed inside of buildings and can be used to relate their signal strength to a certain location;
- Good resolution: through a suitable localization of a set of Bluetooth beacons it is possible to detect distance-depending events triggered when the distance between a device and a beacon is up to some centimetres;
- Low effort of installation: Bluetooth beacons can be easily attached to any object in the environment.

Reeves et al. [18] presented some guiding principles as initial strategies to aid in the development multimodal interfaces. Taking into account such indications we decided to apply the following main design criteria:

- The design structure needs to have different modalities for different contexts;
- To address privacy issues a login process for every user is needed;
- It should be possible to turn off a particular modality, and non-speech alternatives should be provided.

The above mentioned design criteria led to the following requirements:

- Indoor localization: the user current position should be tracked;
- Adaptation of the interaction modality: the system should be able to use the most appropriate modality or combination of modalities for interaction for a certain situation;
- Adaptation of the UI content: the system should be able to select the most appropriate content (information) for a certain situation;
- Independence of different technological spaces: the system should work with different devices, e.g. the HMD on an Android Phone, iPhone, etc.;
- Control Adaptations: it should be possible to add or remove rules for adaptation according to multiple application scenarios.

We decided to follow a user-centred design (UCD) methodology where the needs, desires, and limitations of end users of a product are given extensive attention and which results in a iterative design process. van Velsen et al. [25] described the iterative design process for adaptive and adaptable systems, associated to the goals of User-centred Evaluation (UCE). UCE aims to verify the quality of a product, detect problems and support decisions and find and solve problems in time. The goals of different phases in the iterative design process are shown in Fig. 1.

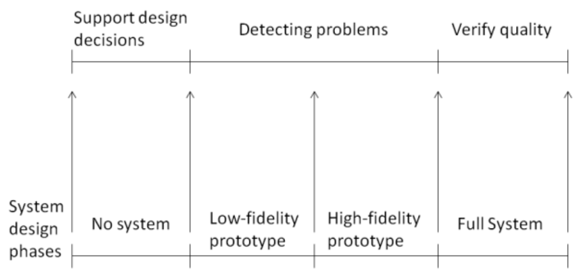


Fig. 2. Phases of the iterative design process [25]

Based on a first user study results of a low-fidelity prototype [3], we identified asset of usability problems specific to the adaptive features of the application and conducted a further user study with an improved high-fidelity prototype. It was concluded that adaptive rules are received well if they trigger the delivery of additional information. However, omitting some information without notification leads to some usability problem. They reported that users were irritated when the visual output was turned off without notification. As a result, the users ranked the respective adaptation rule worse compared to the other rules. In consequence we added the following requirements:

- the users will be informed through visual and vocal output prior to the adaptation;
- the information will explain why and how the adaptation is going to happen.

We believe that this information is crucial to achieve a good user experience when interacting through adaptive user interfaces. To lower the possibility that the user misses the information we decided to add a third modality to the design:

- The user will receive a vibro-tactile feedback prior to the adaptation.

By this additional modality the user will receive notifications on adaptations in a non-intrusive way. User feedback reported in section **Errore. L'origine riferimento non è stata trovata.** evaluates the usefulness of adaptations with and without the support of a vibro-tactile feedback.

4. Software Architecture Supporting the Adaptive Multimodal Application

We have exploited the MARIA model-based framework [15] for obtaining adaptation able to better support various interaction modalities. The framework provides a language for the abstract description (the so-called "Abstract User Interface" level, that defines the UI in a platform–independent manner) as well as multiple platform-dependent languages (which are at the level of the so-called "Concrete User Interface"), which refine the abstract language depending on the interaction resources characterising different platform types. Examples of platforms are the graphical desktop, the graphical mobile, the vocal platform, the multimodal, etc. Moreover, user interface generators for various implementation languages are available starting with such concrete languages. The multimodal concrete language provides a simple and intuitive vocabulary to indicate how to distribute the user interface elements across modalities at various granularity levels.

The current multimodal generator [9] is able to create multimodal HTML 5 applications that rely on the Google support for vocal interaction. Such generated applications are interpreted by an Android native application, which exploits a WebView object and the Android APIs for TTS (Text To Speech) and ASR (Automatic Speech Recognition). Google support for vocal interaction is invoked through JavaScript code and is able to send the user's utterances to a remote vocal recogniser in order to determine the corresponding input string. A HTML5 page generated with this approach consists of two parts: the graphical and the vocal one; the latter is invisible to the users and it contains the definition of output, prompt and vocal feedback as well as the synthesis properties. These parts are linked through the id attributes: the id of the vocal elements are obtained by appending the "_vocal" string to the id of the corresponding graphical elements. Thus, it is possible to obtain different implementations for the same interface element in the two modalities. In order to keep the two versions synchronised, once a graphical element gets the UI focus, the application retrieves its corresponding vocal element (same id + '_vocal' suffix) and invokes the TTS engine. The synthesis properties (e.g. speech, break, emphasis) are represented by a JSON object in the vocal part. If the graphical element is an input, then the application synthesizes a vocal prompt, if available, and starts recording the voice. When a long silence after the vocal input is detected, the user's utterance is sent to the ASR. The result of the ASR processing is then associated with the corresponding graphical element.

Our architecture shows how we can provide support for adaptation through the use of model-based descriptions of interactive applications. At design time the initial version of the application is developed using the MARIA language. In addition, the relevant adaptation rules are specified in terms of events, conditions, and actions according to a language for adaptation rules that will be described later on. Such adaptation rules are triggered by contextual events, which can depend on various aspects (user preferences, environmental changes, application-related events, etc.). The impact of the adaptation rules can have various granularities: complete change of user interface modality (e.g. from vocal to graphical if the environment becomes noisy), change of some user interface parts (e.g. change from map view to order view), or just change of attributes of specific user interface elements (e.g. change of font size).

At run-time we have a server-side Adaptation Engine that is able to communicate with the context manager server in order to receive information on subscribed events and the interactive devices available in order to update the application according to the adaptation rules.

More specifically, the Adaptation Engine has access to the list of the adaptation rules relevant for the current application. Changes in the context, communicated by the Context Manager, may trigger one or more of them. The languages to specify the adaptation rules and the interactive applications are distinct. Thus, it is possible to modify one without having to change the other. However, clear relations among them are defined so that the actions of the adaptation rules can be defined in terms of required modifications to the model-based descriptions of the interactive applications or directly to the implementations.

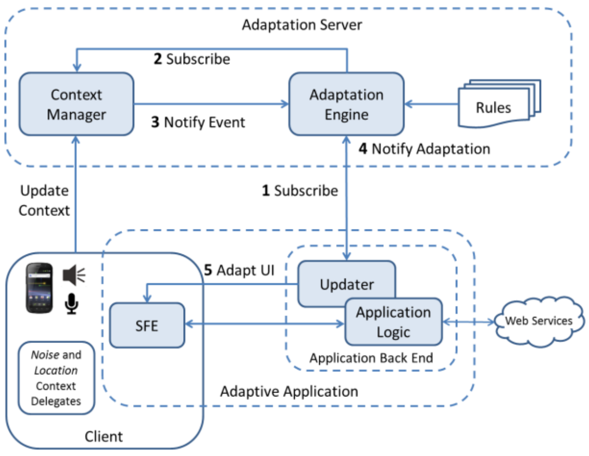


Fig. 2. The software architecture supporting the adaptive application

Figure 2 details the software architecture of the multimodal adaptive application and the run-time support for adaptation. The Adaptive Application structure is depicted on the dashed line box at bottom-right. The application consists of its Service Front End (SFE), directly accessed by the client-side, and on the logic (which depends on one or more external Web Services). A module, named Updater, is responsible for storing adaptations effects (actions) in server side session, which are then forwarded to the client-side scripts that will apply them. An adaptation is defined through a list of modifications of page interactors (i.e. actions to be performed in order to modify the resulting pages) and is actuated by the Adaptation engine according to events detected by the Context Manager and predefined (adaptation) rules. Event detection at run-time is carried out by the Context Manager, which monitors relevant context parameters (which are, in turn, specified in the rules). The Context Manager server is aware of the current context parameter values since it is continuously updated by the Context Delegates running on the client device. In the application considered we have exploited environment noise and user location context delegates.

When the adaptive application is initialized, i.e. at user login time, the Updater subscribes (1) to the Adaptation Engine, in order to be asynchronously notified whenever an adaptation is needed. The Adaptation Engine loads the files of the adaptation rules associated to the application and subscribes (2) to one or more context parameters that are specified in such rules, with the aim to be asynchronously notified (3) by the Context Manager when the rules conditions are satisfied (e.g., when the environment noise level decreases below a certain threshold). Upon reception of an event notification, the Adaptation Engine notifies the Updater about the current adaptation (4). Such a notification is modelled as a list of actions encoded in XML. The Updater manipulates the action list and converts it in a JSON string before sending it to the client-side. The JSON format has been chosen because it is more efficiently manageable by the client browser, i.e. by the JavaScript adaptation script linked in the adaptive application.

The client adaptive application contains a JavaScript that performs the actual adaptations. The updater module sends the actions to the script that parses the JSON action list and, for each item in the list, modifies the actual interface accordingly. A single action can cause the modification of a specific UI element (e.g., a container can be shown/hidden) or of a set of UI elements with certain properties (e.g., all the text input elements are converted into multimodal ones). An action may cause a function call (e.g., the redraw procedure of a map) as well. When the action part of a triggered rule causes a change of modality, then the script adds/removes a corresponding CSS class to/from the considered element. Two CSS classes are introduced for this purpose: the TTS indicates the need of text-to-speech and the ASR indicates the need for the augmented speech recogniser.

The Web adaptive application also includes a script for controlling the level of multimodality. Such script is used to analyse the DOM of the UI and select all the elements with the TTS and ASR classes, which need to synthesize the vocal output or record the user input. Thus, adding or removing one of these two classes results in changes on the multimodal level.

5. Adaptation rules

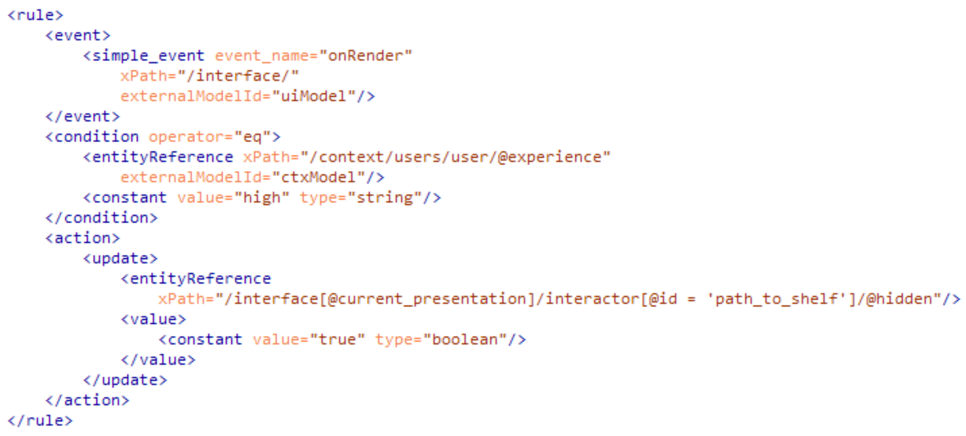
A XML-based high-level description language has been developed to declaratively express adaptations defining the transformations that should be triggered when some specific situations occur in the context (e.g. an entity of the context changes its state) or in the interactive application (e.g. a UI event is triggered). In particular, the three parts of the language are: *event*, *condition*, *action* (ECA). The *event* part escribes the event(s) whose occurrence triggers the evaluation of the rule. This part could specify elementary events occurring in the interactive application, or a composition of events. The *condition* part is represented by a Boolean condition that has to be satisfied in order to execute the associated rule action(s). The condition part is optional. In the *action* part there might be 1 to N simple actions occurring in the interactive application or even 1 to N other adaptation rules. In practice, the action part often contains indications on how the concrete description of the interactive application should change in order to perform the requested adaptation. Event Condition Action is an approach that was originally introduced for structuring active rules in event-driven architectures and active database systems, and has already been used for supporting adaptive user interfaces (see for example [14]). In our case, we have structured it in such a way to easily connect it to the events generated by the Context Manager and the interactive application specification. We are thus able to indicate a wider set of adaptation rules than can be more flexible than those indicated through environments such as IFTTT.com, whose usability limitations have been analysed in [24].

Below we describe a set of example adaptation rules supported by the application developed. For each rule we provide a title with a brief explanation/rationale and the three key parts of its specification (event, condition, action).

- *Fragile object* - The rationale of this rule is that when the worker is about to pick a fragile object, the multimodal UI should switch to only-vocal modality in order not to distract the user while picking the item.
  - Event: a shelf has been reached;
  - Condition: the reached shelf contains a fragile item and the current modality is not only-vocal;
  - Action: Switch from multimodal to only-vocal modality.
- *Picking timeout* - The user has just reached the destination shelf of the item but there is no confirmation of the actual item picking. The application then assumes that the worker is distracted/confused and/or not able to recognize the item to pick, then it provides again info on the item, both graphically and vocally.
  - *Event*: the user has reached the destination shelf;
  - *Condition*: there has not been confirmation of the item picking and the user interface is multimodal;
  - *Action*: the application visualizes an image representing the item to pick, and simultaneously repeats the item name vocally.
- *Order visualization for experienced workers* - If the user has good knowledge of the warehouse shelf organisation, there is no need to show associated path information: the application adapts accordingly.
  - *Event*: beginning of a session with the HMD;
  - *Condition*: the user is a warehouse expert ;
  - *Action*: the application hides the information about how to reach the different shelves.
- *Traffic Jam* - There are multiple workers who are expected to approach the same path at the same time: the application adapts in order to minimise the risk of workers to wait for other people before picking the items.
  - *Event*: order completed;
  - *Condition*: multiple pickers approach the same path at the same time and the path optimization preference is selected;
  - *Action*: the application shows the blocked path suggesting a different route.
- *Noisy environment* – The environment gets noisy, then the multimodal application switches to 'only-graphical' modality.
  - *Event*: the environment gets noisy;

- *Condition*: the noise level is greater than a given value and the application is using both the graphical and vocal modality for interacting with the user;
- *Action*: the application switches to the only-graphical modality.

We show with an example how it is possible to express such adaptation rules through our rules description language. We consider (see Figure 3) the rule for the order visualization for experienced workers. When the interaction starts (the presentation raises the onRender event), if the current user has high knowledge (encoded as an attribute in the 'user' part of the context model), then the application hides the path to reach the shelf referred by an interactor with id path_to_shelf, by setting its hidden attribute to true. A symmetrical rule manages the case of inexperienced workers.



```
<rule>
    <event>
        <simple_event event_name="onRender"
            xPath="/interface/"
            externalModelId="uiModel"/>
    </event>
    <condition operator="eq">
        <entityReference xPath="/context/users/user/@experience"
            externalModelId="ctxModel"/>
        <constant value="high" type="string"/>
    </condition>
    <action>
        <update>
            <entityReference
                xPath="/interface[@current_presentation]/interactor[@id = 'path_to_shelf']/@hidden"/>
            <value>
                <constant value="true" type="boolean"/>
            </value>
        </update>
    </action>
</rule>
```

Fig. 3. Formalization of the Order visualization for experienced workers rules

6. The Resulting Adaptive Application

The resulting application for context-aware and guided picking in warehouses addresses the requirements from the previous section. The system uses three modalities – visual, voice and tactile. The use of the visual modality is based on a head-mounted display (HMD) connected to a smart phone. It allows head-up navigation, hands-free interaction and augmentation of the reality. We use a voice output and input module that allows the user to receive and submit information while maintaining the hands-free paradigm. The tactile feedback is based on the vibration feature of the Smart Phone worn on the user's arm. In exceptional cases where hands-free input (i.e. vocal) is not possible (e.g. noisy environment) the touchpad of the Smart Phone is used.

The user logs in and is tracked by an indoor-positioning system based on small Bluetooth beacons placed in the environment (i.e. at the shelves). Localization consists in detecting when the user is in proximity of a point of interest. The noise is measured by the microphone of the smartphone. The information on the user's identity, position and the noise level is sent to the Context Manager to allow adaptation of modality, screens and content. The adaptability of the UI opens up the possibility to work with multiple interaction modalities i.e. vocal, visual and tactile. The UI adapts by reacting to changes in the context (e.g. noise level of the environment) and choosing suitable combination of modalities for interaction.

The visual output is based on a HMD, the vocal output exploits a text-to-speech engine, the vocal input relies on Google support for vocal interaction and the vibro-tactile feedback uses the vibration feature of the smartphone as shown in Fig. 4.

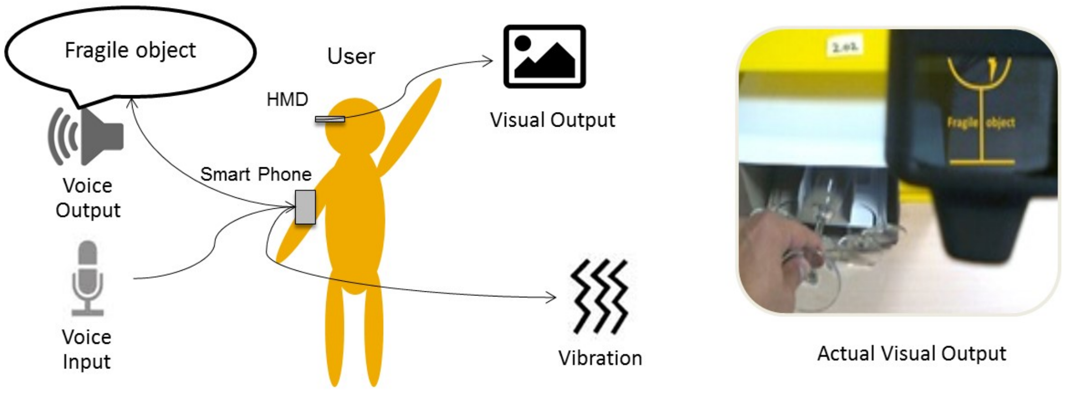


Fig. 4. Interaction Modalities and devices used in the application developed & an actual Visual Output on HMD

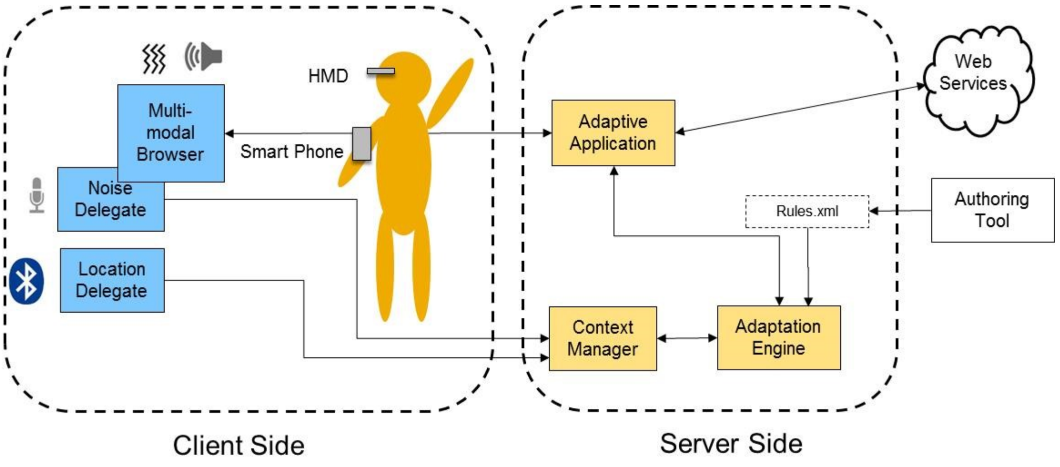


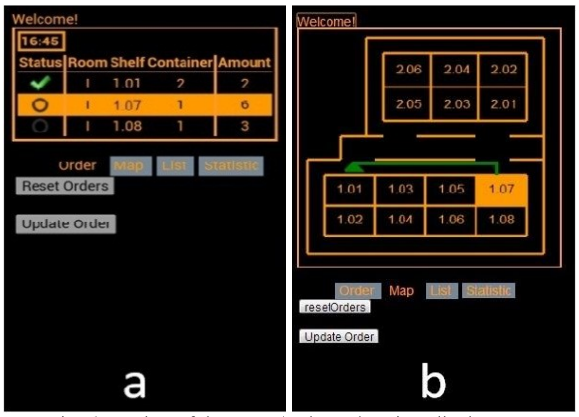
Fig. 5. Overall Architecture: The client-side parts (Multimodal application, Noise Delegate and Location Delegate) communicate with the server-side parts (Context Manager and Adaptation Engine)

Figure 5 relates the software architecture (Figure 2) supporting the adaptive application to the interaction modalities and devices (Figure 4).

In this case study we have developed and deployed three mobile components, two of which are processing sensor data (context delegates) and the multimodal application. The Location Delegate application supports an indoor positioning system using several mini Bluetooth-beacons. It scans the surrounding environment looking for these Bluetooth-beacons, then simply sends information on all detected beacons to the CM (Context Manager) server, which identifies the closest beacon. Since each shelf and some additional positions are equipped with one beacon, discrete positions can be determined through this solution. The second mobile application is the Noise Delegate, which uses the microphone-sensor of the mobile device to detect the noise level in the area and sends this information to the CM.

We have developed an Android Application that encapsulates a WebView object enriched with some scripts in order to obtain the multimodal adaptive Web application. Such object displays the user interface generated and the corresponding dynamic adaptations results generated depending on the current user's environment. The inserted scripts are able to exploit Android API for Text-To-Speech (TTS) and Automatic Speech Recognition (ASR). The Adaptation Engine triggers the change in how to use the modalities (speech, vibro-tactlile and visual) for obtaining various adaptations in the application for efficient picking in various scenarios.

The Graphical User Interface (GUI) is structured into four main screens presenting the business data. These are Order, Map, Task and Statistics, but for the sake of brevity only the Order view and the Map view are discussed.


Welcome!

16:45

Status	Room	Shelf	Container	Amount
✓	I	1.01	2	2
○	I	1.07	1	6
○	I	1.08	1	3

Fig. 6. Design of the GUI a) The order view displays status, shelf number and amount of items for the previous, current and next pick. b) The Map view displays the 2-D map of the warehouse. The path is indicated by a green line from the current location (1.01) to the destination shelf (1.07)

The Order view (see Fig. 6.a) is a tabular view representing a part of the order yielding information on the previous, current and next pick. It shows the status, the location (i.e. room, shelf and container number) and the amount to be picked. The previous pick is marked with a green tick, while the current and the next pick show an empty circle. The Map view (Fig. 6.b) reflects a simplified 2-D (bird's-eye) view of a warehouse with two rooms each having two doors and connected through a corridor. The rooms contain 6 and 8 shelves respectively. The recommended path to follow is indicated by a green line, starting from the current position marked by a green arrow (i.e. 1.01) to the destination shelf for the current pick whose box is highlighted in inverted colour (i.e. 1.07).

The basic interaction flow shown in Fig. 7 provides a brief insight on the steps of the business process.

At start-up the system recognizes the current location of the user and displays the Order view. The Order view consists of the order details including room, shelf and amount number with the current pick highlighted (see Fig. 7a). After some time (about 10 seconds) the view switches automatically to the Map view indicating the user's current location and the pathway to the destination shelf (see Fig. 7b). The user's current location is constantly updated based on the information from the Bluetooth-delegate (see Fig. 7c – d). After reaching the destination shelf the user is asked to confirm the shelf number (see Fig. 7e). The user then confirms the shelf number through vocal input. While repeating the user's input the system announces the amount number and asks for a confirmation.

Some situations require UI adaptation according to the rules presented in the previous section and managed by the Adaptation Engine. As an example the resulting interaction sequence for the scenario "Picking a fragile object" is shown in Fig. 8.

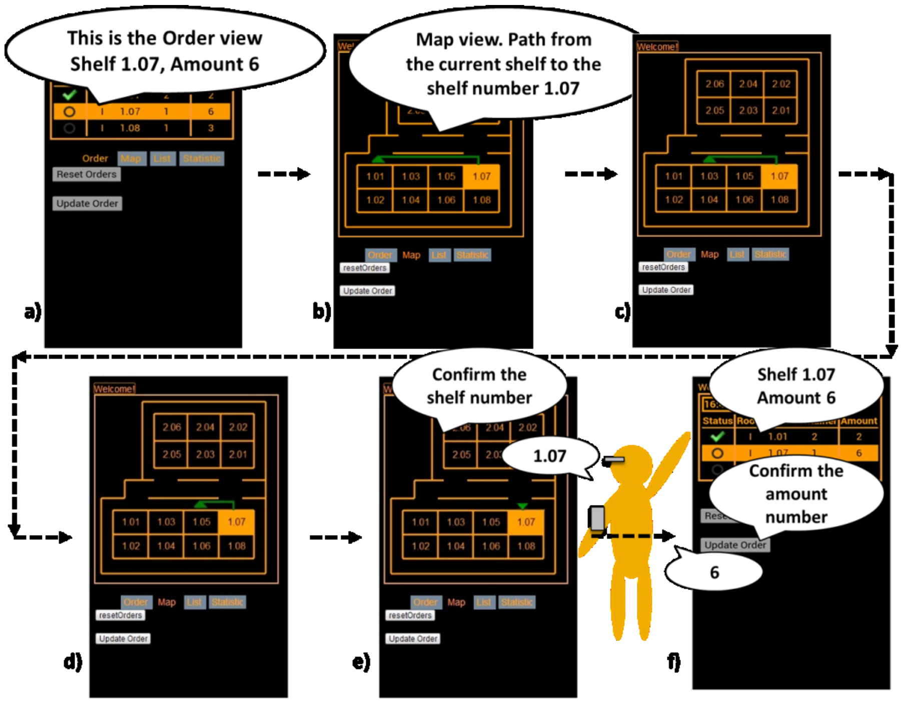

This is the Order view Shelf 1.07, Amount 6
Map view. Path from the current shelf to the shelf number 1.07
Confirm the shelf number
1.07
Shelf 1.07 Amount 6
Confirm the amount number
6
a)
b)
c)
d)
e)
f)

Fig. 7. Basic interaction flow: a) start – system provides overview through Order view. b-d) after 10 seconds – guided navigation through Map view. e). User reaches destination – system asks for shelf confirmation - user confirms shelf. f) system switches to Order view - asks for amount confirmation - user confirms amount. Callout boxes represent vocal interaction

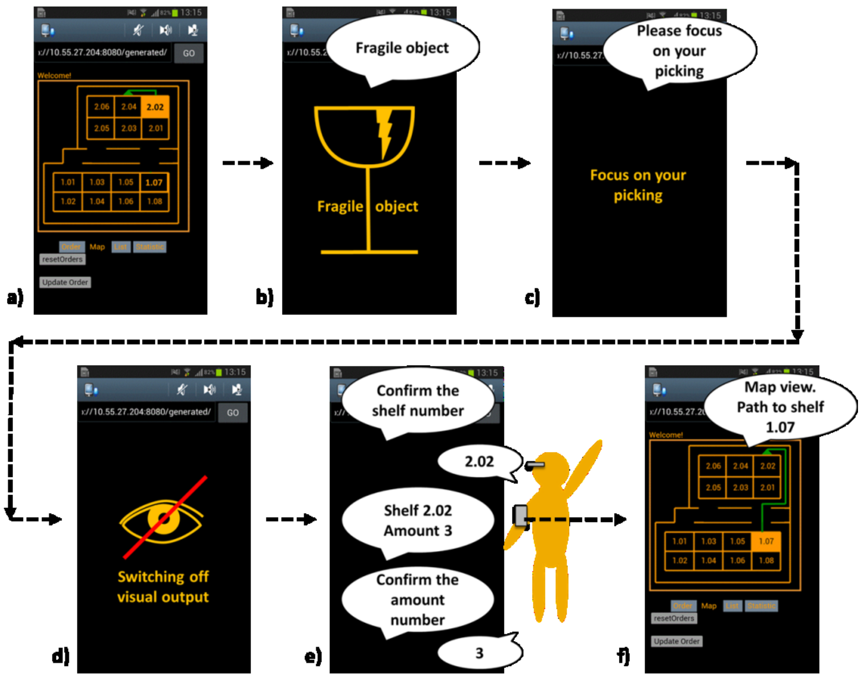

Fragile object
Please focus on your picking
Focus on your picking
Switching off visual output
Confirm the shelf number
2.02
Shelf 2.02 Amount 3
Confirm the amount number
3
Map view. Path to shelf 1.07
a)
b)
c)
d)
e)
f)

Fig. 8. Adaptation flow for Fragile Object: a) Guided navigation b) User reaches destination – system informs user about fragile object c) systems displays advice d) system informs about adaptation action e) interaction proceeds with vocal modality only f) system removes adaptation

The user has reached the destination shelf through guided navigation. At this time the user is informed that the object to be picked is fragile (see Fig. 8b) followed by a warning that care is required (see Fig. 8c) and as a consequence the visual modality will be switched off (see Fig. 8d). It can be seen that the adaptation is preceded by a warning of the current situation and the imminent adaptation action. The need for this was suggested in preliminary user studies [3]. Additionally, the vibro-tactile feedback has been used to announce the upcoming adaptation. As shown in Fig. 8e only the vocal interaction is then used. After the pick is completed the system switches back to multimodal interaction as shown in Fig. 8.f. The following table (Table 1) shows the three examples of context variation and the adaptations triggered by the corresponding rules in the system.

Table 1. Variations of the context and its consequences for the interaction modalities

Context variation	Interaction consequence
The items to be picked are fragile	After confirmation of arrival at the destination by the picker, the visual output will be switched off, only vocal remains.
The route is blocked	The Map view marks the blocked path and suggests an alternative route. (Fig. 9 a)
The environment is noisy	The vocal input and output is switched off, only visual output remains. (Fig. 9 b)

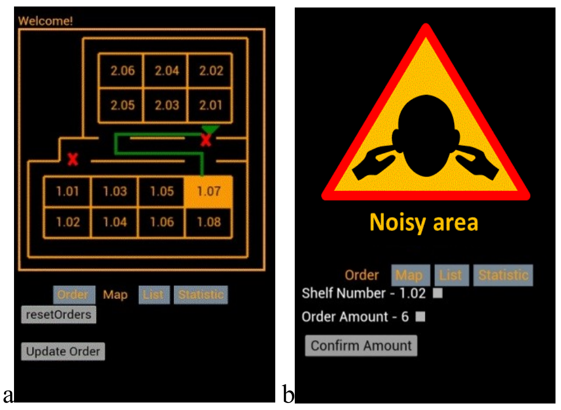


Fig. 9. Screenshot of additional adaptations: a) Route is blocked – alternative route is suggested b) Environment with high noise – Vocal modality is switched-off

7. User feedback

The goal of the user study was to evaluate if the usability related to the adaptive user interfaces can be improved by providing additional information and vibro-tactile feedback prior to the adaptation.

We compare our findings with the results of Bongartz et al. [3] who used a low-fidelity prototype describing the effects of the five adaptation rules from the end-users point-of view. In that case the prototype used a paper-based map to simulate the warehouse layout. Walking and picking of the participants were realized as hypothetical actions. The participants were asked to comment their actions, e.g. by saying “I walk to shelf 473 now”. Also the participants were only informed about a certain situation, e.g. “Imagine you are now in a noisy environment”.

In this new user study the participants were able to walk through the rooms and pick items from boxes. Situations like the sudden occurrence of noise and an obstacle blocking a door were realized through devices and objects. As an additional feature compared to [3] the participants were also able to perceive their movements through the changing pathways in the display.

Besides using a high-fidelity prototype the goal of this user study was to evaluate whether the comparably poor performance of the rules Fragile Object, Experience User and Noisy Environment reported in [3] was improved by providing additional information (i.e. also called user support in [16]) before showing the adaptation in UIs.

Finally we evaluated the usefulness of a vibro-tactile feedback to raise the awareness for the information prior to the adaptation.

The general concept “usability” in this case was operationalized by several more specific aspects, e.g. level of difficulty of the task, system support for the task and usefulness, which were assessed by a questionnaire. We decided not to evaluate measures of efficiency here. We believe that wearable technologies supported by adaptive UIs first need to take the hurdle of usability to proceed to the next level of adoption. Once this level is reached a pilot installation in a central distribution centre running for some weeks is well suited to get reliable measures of efficiency.

To address the user satisfaction, the three adaptation rules (with and without vibro-tactile support) were the independent variables. We had a within-subject design, meaning that every participant was confronted with every adaptation rule with and without vibro-tactile support. The dependent variables were the subjectively perceived user satisfaction of the adaptation rule as assessed in a four-item questionnaire:

1. Navigating to the shelf: This task was [very difficult 1 … very easy 7];
2. Navigating to the shelf: The system support for this task was [very bad 1 … very good 7];
3. Picking the items correctly from this shelf: This task was [very difficult 1 … very easy 7];
4. Picking the items correctly from this shelf: The system support for this task was [very bad 1 … very good 7].

We asked to rank between 1 and 6 general aspects such as likability, usefulness and pleasantness:

1. Fragile object adaptation with vibration;
2. Fragile object Adaptation without vibration;
3. Blocked route adaptation with vibration;
4. Blocked route adaptation without vibration;
5. Noisy environment adaptation with vibration;
6. Noisy environment adaptation without vibration.

Participants were company staff members or students of the local university. A total of 11 participants took part in the study, 8 were male and 3 were female. The average age was 34 years (SD = 8.8). The technical set-up consisted of a HMD connected to a Smart Phone worn on the upper arm. The HMD showed the GUI while the vocal interaction was performed directly through the Smart Phone as shown in the previous section. The interaction sequence was determined by the system where the execution of an adaptation rule was triggered by some change in the context of use (e.g. the user position).

Participants were first introduced to the scenario, the devices and the interface in order to become familiar with the warehouse application, and to wear the devices and interact with the interface. Participants were asked to first conduct a test run by performing a sequence of tasks that started with the systems requesting to pick items from a certain shelf (1.07), and then requiring the user to walk to that shelf, collect the items in a trolley (see Fig. 10) and ended with the user confirming the amount.

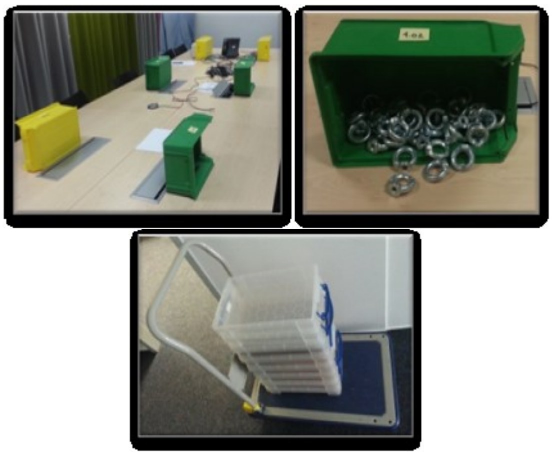


Fig. 10. Multiple shelves (top-left), single shelf (top-right) and trolley (bottom)

After ensuring that the participants understood the basic interaction flow of the interface, the first run continued seamlessly considering the three context situations: fragile object (at shelf 2.02), blocked route (pathway between shelf 2.02 and 1.07) and noisy environment (in the vicinity of shelf 1.02) and the respective adaptations. After each adaptation a 4-item questionnaire was filled out. In between these three context situations “normal” picks following the basic interaction flow also needed to be done. Upon reaching the final shelf (1.08) the vibro-tactile modality was toggled, i.e. if vibration was off in the first run then it was turned on and vice-versa. Then, the second run started following a different pathway but with the same adaptations as before. Again a four-item questionnaire was filled out after each adaptation. After reaching the final shelf (1.08) the participants ranked the 2x3 adaptations. Notes on the ranking for the three aspects (i.e. general likability, usefulness and pleasantness of the adaptation) were taken in order to gather further information.

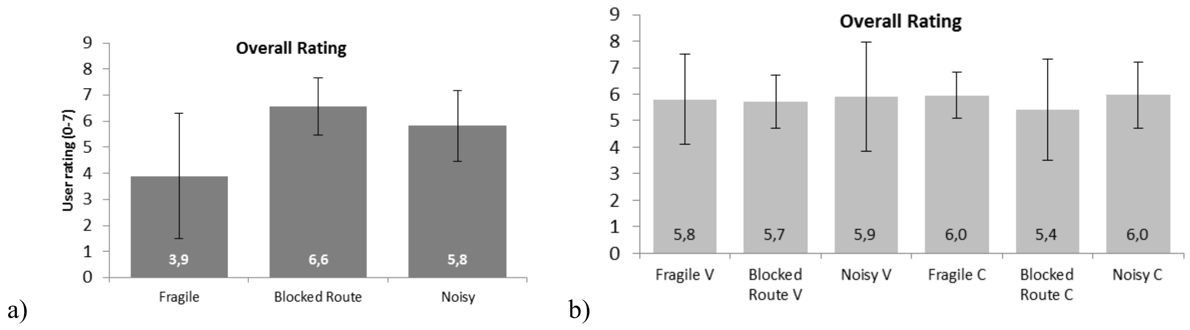


Fig. 11. Overall rating of adaptation rules for Fragile Object, Blocked Route and Noisy Environment: a) from the user study on the low fidelity prototype [3]; b) from the last user study on the high-fidelity prototype with (V) and without (C) vibro-tactile feedback

We compared the result for the aggregated overall rating from the user study on the low fidelity prototype [3] with the current user study on the high fidelity prototype. Diagrams in Figure 11 show that the difference in ranking between the adaptation rules in the last reported user study (Figure 11 b) is much smaller compared to the first user study (Figure 11 a). The current user study shows an average rating between 5.4 and 6.0 on a scale from 0 (most negative) to 7 (most positive) while the first user study shows an average rating between 3.9 and 6.6 on a scale from 0 (most negative) to 7 (most positive). We believe that this is due to the fact that the user satisfaction was improved by adding information prior to showing the adaptation in UIs. In the current prototype the screen display is not simply switched off as reported in [3]. First the user is informed about the situation "Fragile Object" which gives the reason for the adaptation (see Fig.8). Then optionally a recommendation is given, i.e. "Focus on your picking". Finally the user is informed about the adaptation action, i.e. "Switching of visual output".

The result for the aggregated overall rating of the current study (see Figure 11b) does not indicate any difference between the adaptations variants with/without vibro-tactile feedback. The delta of the ratings within the variants range from 0,1 for Noisy Environment to 0,3 for Blocked Route which is marginal given the variance. In order to achieve a finer granularity we used an ordinal ranking exercise. Participants were not allowed to assign the same ordinal rank to more than one variation. Except for the adaptation rule Noisy Environment in the Preference scale (see Fig. 12), all rules were ranked better with vibro-tactile feedback than without for all ranking scales (see Fig. 13 and Fig. 14). Some participants reported that in the noisy situation the impact of the environment (jack-hammer noise) was already so intense that they preferred less intensive feedback from the UI. The diagram (see Fig. 13) shows that the participants found the adaptation for the noisy environment less useful. This might stem from the fact that in this situation the users had to confirm the amount of items by pressing the touchpad of the Smart Phone. Locating and pressing the button was often commented as being tedious. In this situation the benefits of hands-free interaction are lost.

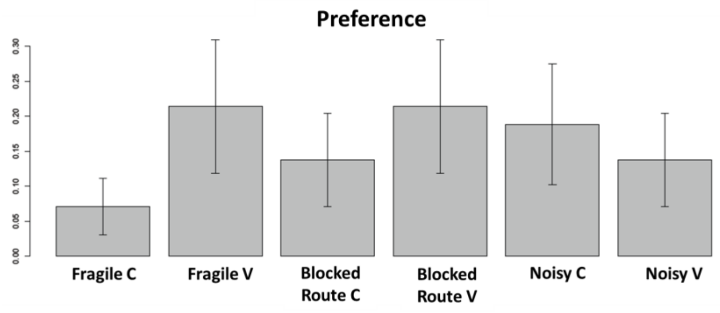


Fig. 12. Results from the ordinal ranking exercise for the Preference criterion for each adaptation with (V) and without (C) vibro-tactile feedback

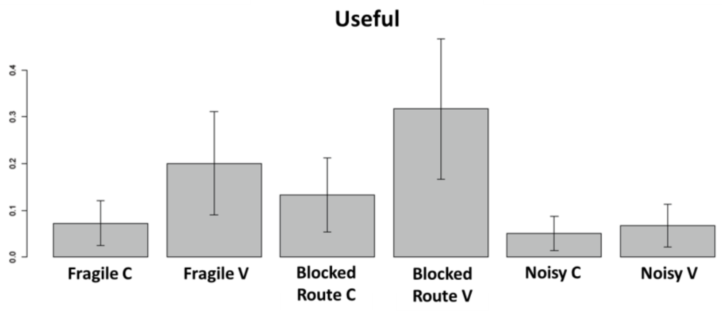


Fig. 13. Results from the ordinal ranking exercise for the usefulness criterion for each adaptation with (V) and without (C) vibro-tactile feedback

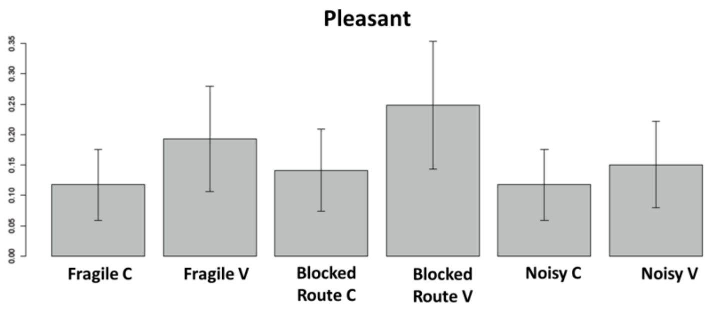


Fig. 14. Results from the ordinal ranking exercise for the pleasantness criterion for each adaptation with (V) and without (C) vibro-tactile feedback

In the free-form answers of the questionnaire the participants highlighted that the map view, the automatic switching between screens and the vocal output were very helpful. The system was perceived as very supportive in the tasks of navigation and picking. Some participants criticized the voice interaction for its timing: while some outputs took too long, some inputs were triggered too early. Some participants reported that they were not aware that a vibration was preceding the adaptations. Additionally, many participants reported problems due to the fact that the map view did not align with the direction of the gaze. They also reported the HMD is difficult to adjust during the preparation phase, and we observed that some participants needed to re-adjust the HMD during the run.

8. Conclusions

User interfaces that change the interaction modalities according to contextual aspects can be of great benefit in the work domain. Intelligent environments able to provide such adaptive user interfaces are thus highly desirable. In this paper we have reported on a solution that relies on model-based descriptions of an interactive application to facilitate generation and dynamic update of different versions of the application depending on the context of use. Thanks to such logical descriptions, different application versions can be generated and users can benefit from the various interaction modalities. The policies for exploiting such modalities at run time are defined in the adaptation rules, which are separate (and thus can be modified independently) from the UI definition. With the presented solution we support adaptation at various granularity levels, ranging from the total change of the interactive application version due to a change in the interaction modality to minor modifications of the current version.

So far we have considered a warehouse picking case study, indicating how a set of integrated tools (context manager, adaptation engine and predefined adaptation rules) have been combined, and we have also reported the results of a usability evaluation of the integrated solution. We have adopted a HMD-based device (Smart Glasses) because it supports the scenario with head-up navigation, hands-free interaction, and reality augmentation. These devices require adaptive user interfaces that change the interaction modalities taking into account contextual aspects. We have also chosen to use Bluetooth beacons as they can be used indoors, have a high resolution and a low effort of installation. However, the solution that was applied to the user study with roughly 14 distinct locations (shelves) may pose considerable challenges when scaling to a large central distribution warehouse. The investment cost of providing thousands of shelves with Bluetooth beacons would rather high. A likely cost-effective solution would be a mixed system with some locations equipped with Bluetooth beacons while others rely on different technologies.

We followed a user-centred design (UCD) methodology to develop a high-fidelity prototype which addressed the problems detected during the previous user study on a low-fidelity prototype. Our user study showed that informing the user before the adaptation by giving prior visual and vocal explanation of how and why the adaptation is going to take place raises user satisfaction. The study also showed evidence that notifying the user with a vibro-tactile feedback prior to the adaptation generates a positive user experience. We argued that wearable technologies supported by adaptive UIs first need to overcome the hurdle of usability to proceed to the next level of adoption, i.e. efficiency.

Future work will aim to improve the engineered part of our solution by e.g. extending the context-aware capabilities.

Concerning the industrial application, further scenarios will be investigated. One prominent example will be the task of a service technician, which includes locating a faulty object and obtaining useful information on the way to replace or repair it. While working on this scenario we also plan to tackle also the usability issues that were reported during the user study. On the hardware side, the recently emerging Smart Glasses will be tested to solve issues such as the adjustment of the display, and wired connection between display and smart phone. We are also planning to investigate the merger of different sensor technologies, i.e. Bluetooth, Wifi and GPS. We are confident that the problems stemmong from the use of a single technology (e.g. high investment costs) will be overcome and allow for a seamless user experience from an outdoor location to an indoor location. On the GUI side we aim for a representation of the map taking into account the users' perspective. In addition, we plan to improve the performance of the system in the terms of response-time, to apply the presented approach to other case studies and to run a study targeting designers and developers. The latter step will be devoted to better assess to what extent the model-based approach may facilitate the development of adaptive applications.

Acknowledgements

This work has been supported by the SERENOA EU ICT Project, http://www.serenoa-fp7.eu/

References

- S. Ali, A. Lewandowski and J. Rett,A SOA based context-aware order picking system for warehouses using Laser Range Finder and wearable computer, in: *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM),* pp. 1-8. (2011).
- J. McNaull, J.C. Augusto, M. Mulvenna and P. McCullagh, Flexible context aware interface for ambient assisted living, *Human-centric Computing and Information Sciences*, 2014, 4:1, Springer.
- S. Bongartz, Y. Jin, F. Paterno, J. Rett, C. Santoro and L.D. Spano, Adaptive User Interfaces for Smart Environments with the Support of Model-based Languages, in: *Proc. of AmI 2012*, vol. 7683 pp. 33-48. Springer.
- J. Coutaz, L. Nigay, D. Salber,.A. Blandford, J. May and R. Young, Four Easy Pieces for Assessing the Usability of Multimodal Interaction: the CARE Properties. in: *Proc. of INTERACT 1995*, pp.115-120.
- C. Duarte, and L. Carriço, A Conceptual Framework for Developing Adaptive Multimodal Applications, in: *Proc. of IUI'06*, ACM, 2006, pp. 132-139.
- B. Dumas, D. Lalanne and S. Oviatt, Multimodal Interfaces: A Survey of Principles, Models and Frameworks, *Lecture Notes in Computer Science*, Volume 5440, 2009, pp 3-26, Springer.
- M. Honkala, and M. Pohja, Multimodal interaction with XForms, in: *Proc of ICWE 2006*, pp. 201-208.
- J. Lawson, A. Al-Akkad, J. Vanderdonckt and B. Macq, An open source workbench for prototyping multimodal interactions based on off-the-shelf heterogeneous components, in: *Proc. of EICS 2009*, ACM, pp. 245-254
- M. Manca, F. Paternò, C. Santoro and L.D. Spano, Generation of Multi-Device Adaptive MultiModal Web Applica-

tions, in: *Proc. Of MobiWIS 2013*, LNCS N.8093, pp.218-232, Springer Verlag, August 2013.

– E. Marcotte, Responsive Web Design, *A Book Apart*, 2011, http://www.abookapart.com/products/responsiveweb-design

– G. Meixner, F. Paternò and J. Vanderdonckt, Past, Present, and Future of Model-Based User Interface Development, i-com 10(3): 2-11 (2011)

– M. Nebeling, M. Speicher, and M.C. Norrie, W3Touch: Metrics-based Web Page Adaptation for Touch, in: *Proc. of CHI 2013*, ACM, 2013, pp. 2311-2320.

– Z. Obrenovic, D. Starcevic and B. Selic, A Model-Driven Approach to Content Repurposing, *IEEE Multimedia*, January March 2004, pp.62-71.

– J. Octavia, L. Vanacken, C. Raymaekers, K. Coninx and E. Flerackers, Facilitating Adaptation in Virtual Environments Using a Context-Aware Model-Based Design Process, in: *Proc. of TAMODIA 2009*, LNCS 5963, pp.58-71.

– 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. Computer-Human Interaction*, 16(4), 1-30. (2009, Nov.).

– T.E. Paymans, J. Lindenberg and M. Neerincx, Usability trade-offs for adaptive user interfaces: ease of use and learnability, in: *Proc. of IUI '04*, ACM Press (2004), 301-303.

– S. Raiyani and J.M. Kumar, Multimodal warehouse application, *Interactions Managize*, (July 2006), 34-37. DOI=10.1145/1142169.1142193 http://doi.acm.org/10.1145/1142169.1142193

– L.M. Reeves, J. Lai, J.A. Larson, S. Oviatt, T.S. Balaji, S.P. Buisine, P. Collings, P. Cohen, B. Kraal, J.-C. Martin, M. McTear, T. Raman, K.M. Stanney, H. Su and Q.Y. Wang, Guidelines for multimodal user interface design, in: *Communications of the ACM* 47(1), pp. 57--59 (2004).

– D. Salber, A. Dey and G. Abowd, The context toolkit: Aiding the development of context-enabled applications, in: *Proc. of CHI'99*, ACM, 1999, pp. 434-441.

– B. Schwerdtfeger and G. Klinker, Supporting Order Picking with Augmented Reality, in: *7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pp. 91-94. (2008)

– J.S. Sottet, V. Ganneau, G. Calvary, J. Coutaz, A. Demeure, J.M. Favre and R. Demumieux, Model-Driven Adaptation for Plastic User Interfaces, in: *Proc. of INTERACT 2007*, pp. 397-410

– A. Stanciulescu, Q. Limbourg, J. Vanderdonckt, B. Michotte and F. Montero, A Transformational Approach for Multimodal Web User Interfaces based on UsiXML, in: *Proc. of ICMI* 259-266 (2005)

– S. Sire and C. Chatty, The Markup Way to Multimodal Toolkits. In: *W3C Multimodal Interaction Workshop* (2004) [http://www.w3.org/2004/02/mmi-workshop/sire-intuilab.html]

– B. Ur, E. McManus, M.P.Y. Ho and M.L. Littman, Practical trigger-action programming in the smart home, in: *Proc. of CHI 2014*, ACM, 2014, pp. 803–812.

– L. van Velsen, T. van der Geest, R. Klaassen and M. Steehouder, User-centered evaluation of adaptive and adaptable systems: *A literature review, Knowl. Eng. Rev. 23*, 3 (2008), 261-281

– K.A. Weaver, H. Baumann, T. Starner, H. Iben and M. Lawo, An empirical task analysis of warehouse order picking using head-mounted displays, in: *Proc. of 28th international Conference on Human Factors in Computing Systems (CHI`10)*, ACM, New York, NY (2010)

– T. Yang, M. Ferati, Y. Liu, R.R. Ghahari and D. Bolchini, Aural Browsing On-The-Go: Listening-based Back Navigation in Large Web Architectures, in: *Proc. of CHI 2012*, ACM, 2012, pp.277-286.