# Building Supportive Multimodal User Interfaces

**José Coelho**
LaSIGE, University of Lisbon
Campo Grande Edifício C6 Piso 3 1749-016
Lisboa, Portugal
+351 21 750 05 32
jcoelho@lasige.di.fc.ul.pt

**Carlos Duarte**
LaSIGE, University of Lisbon
Campo Grande Edifício C6 Piso 3 1749-016
Lisboa, Portugal
+351 21 750 05 19
cad@di.fc.ul.pt

**ABSTRACT**
In this paper, we describe and discuss solutions capable of helping in the development of supportive multimodal user interfaces. Based on the specifications and design of European Union funded project GUIDE (Gentle User Interfaces for Elderly People), we show how it is possible to use several modalities of interaction as well as adapting UIs, as a mean of providing users with ideal interaction in every application, and preventing or resolving errors resulting from missed or wrong user-device inputs.

**Keywords**
Supportive multimodal user interfaces, adaptation, GUIDE, UI translation.

**INTRODUCTION**
In this paper we are going to introduce some mechanisms present in the ongoing GUIDE project and which are intended to help developers in the implementation of supportive user interfaces.

**GUIDE Project**
GUIDE[i] aims to offer multimodal interaction to elderly (and disabled) users with the goal of simplifying interaction with a television (TV) and set top box (STB) based system. By pointing to the screen, making gestures, issuing speech commands, interacting with a Tablet PC, using the remote control, interacting with an Avatar or simply making use of user intuition for combined interaction with several of these modalities, the GUIDE framework makes fitting interaction to users' characteristics and preferences, possible and also for impaired users to interact with the TV.

In what concerns supportive interaction, the use of Avatars is explored with the goal of offering users, a persona with whom they can relate to, while interacting with the system. The Avatar will work like someone who explains to users the interaction steps to be done in order to execute tasks, and will help them getting out of "trouble" after an error has been generated while using the system. More, the existence of generic, as well as content-specific, speech commands as a possibility of interaction makes intuition a reality in GUIDE. Additionally, pointing interaction using a video based gesture tracking sensor is helped by cursor adaptation techniques which makes easier the selection of content on the screen, also helping in supporting interaction.

This diversity of devices and modalities of interaction, will offer users the flexibility to use whatever medium they find more appropriate given a specific context, at the same time as they benefit from visual (text, images, video and animations), audio (speech, and other sounds) and haptic feedback (vibration). These multimodal capabilities are in fact, the first step to a supportive interaction.

Considering the variety of differences present in elderly users and their preferences when using a system like this, GUIDE will cluster it's users in different User Profiles (UPs) - transparent to every user - where data concerning preferences and constraints of interaction are saved. By making use of each UP, GUIDE will try to adapt User Interface (UI) elements to fit every user.

In addition to providing supportive use, GUIDE framework supports UI adaptation for every application running. Moreover it aims at providing this support requesting reduced extra effort from developers. Since it is not expectable to have developers providing different versions of applications for users with different characteristics GUIDE will develop tools to "translate" a "standard" UI into tailored UIs for every type of user. The extra effort asked of developers consists in identifying each UI interactive component using WAI-ARIA[ii] semantic tags. With that information, GUIDE will abstract UI characteristics, and save them in an Application Model (AM) (one for every application), making adaptation of UI components possible at run-time.

**Problem Description**
Nowadays, most UIs lack capability in guiding users to an adequate and efficient interaction [1], when ideally "the UI must guide the user in accomplishing a task the application was designed for" [4] by providing help and appropriate feedback about features, tasks, modalities and contexts of interaction. If a user is not capable of perceiving an application and reacting to errors while interacting, more sooner than later he or she is going to abandon its use, and adopt a more usable application. Unsatisfied users are going to prefer a better supportive interface which can fit

and adapt to his or her characteristics. If this is true for the so called typical users, for elderly users this is even more relevant. Because these users are usually characterized by having one or multiple impairments (example: hearing difficulties, visual incapacity, motor constraints, etc.), adequate interaction is only possible when the system is capable of adapting its UI components and modalities of interaction to these users' specific characteristics.

Therefore, in the development of supportive multimodal user interfaces for elderly or impaired users, several questions need to be answered so that an appropriate application and interaction can be implemented:

- How to let your users know how to interact?
- How to know your users?
- How to help users after a mistake has been identified?
- How to present content and interaction possibilities in the most suitable way to the users?

In the remainder of this paper, we describe the approaches followed in GUIDE to try to offer solutions to the questions identified above, by supporting multimodal interaction and UI adaptation for elderly users when using a TV and STB based system. Special interest also goes to the way this framework provides every application with the possibility of adapting to different contexts of interaction, and to the presentation of ideas on how it could be possible for these types of users to personalize UI presentation and interaction while preserving usability.

## ANSWERING THE QUESTIONS

### How to let your users know how to interact?

For an efficient interaction to be a reality, users need to have knowledge about the available ways for performing each task. They have to know to the full extent all the possibilities and modalities when confronted with different difficulties and contexts of interactions. Only by understanding how they can interact, they can make the most of the interface being presented and understand how to use all the features provided by the application. For example, if a visual interface with a menu is presented on the screen, and the user doesn't know he or she can speak the name of a specific button for making a selection, a lot of time can be lost by performing the task using alternative modalities (the only ones the user has knowledge about) like selecting the button by pressing remote control keys in a certain sequence or by pointing to the screen with the remote control.

GUIDE will try to instruct the users before they start interacting with any of the framework applications. For this, it will use an application called the User Initialization Application (UIA) to give the user a clear understanding of the possible ways of interaction. Users will be guided through the experimentation of the various modalities of interaction available in the framework, like pointing to the screen, issuing speech commands, pressing remote control buttons, etc.. For this purpose, the UIA will present on the screen a tutorial with scripted animations of how to perform different gestures, informing the user of the set of speech commands he or she can issue for achieving typical tasks, and providing instructions about how to interact with other components of the system like the Avatar engine, the Table PC, etc.. In all this process the user has an active role, learning by experimentation of every interaction modality and device.

### How to know your users?

For the users to understand an interface and know how to interact with it, it really helps that the interface knows the user in advance. Only knowing beforehand what are the users preferred ways of interacting as well as the users' impairments and difficulties makes it possible to build or adapt the interface for appropriate and efficient user interaction. For example, if the system doesn't have enough information about the user to know that he or she is blind and presents a visual interface to him or her, no interaction will occur at all, and the system will not be used. In a second example, if the user prefers to interact using pointing and the system presents a simple visual interface that only receives remote control input, he or she will be less motivated to use and adopt that system (and a higher probability of making errors during interaction exists).

GUIDE will try to collect information about its users before they get to interact with any of the system's applications. To this end, the UIA will also be used for collecting data about users. Every time a new user starts using the system, the UIA is presented on the screen combined with audio output (covering possible situations of severe audio or visual impairments) and the user is asked to perform a series of tasks concerning his or her capabilities. In a first instance, the user is "registered" in the application using name, and facial and vocal characteristics, so that from that point on, every time he or she wants to use the system the correspondent UP can be loaded based on these properties. Next, the application tries to understand if the user has some visual impairment by presenting text on the screen and asking for user feedback (figure 1) (e.g. presenting a sentence and expecting for user to adjust the font until he feels comfortable reading, and then asking user to read the sentence out loud to make sure he is in fact seeing it well). If the user passes this test, different configurations of text font and buttons, as well as several background and text colors, are tested out to understand his or her preferences regarding visual interfaces. If the user fails the test, the text font size is raised in a screen-by-screen basis until there is the understanding of how severe is the user visual impairment.

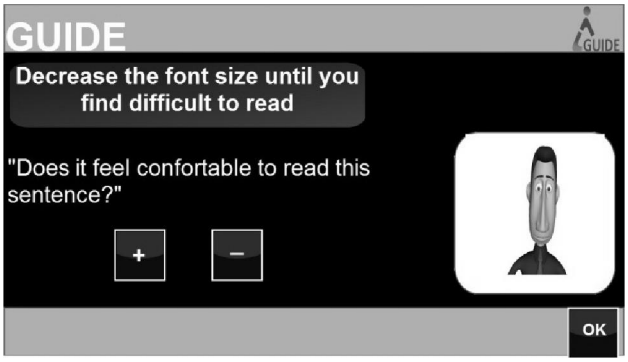


Figure 1: *UIA prototype. Example of visual test where the user has to read out loud the text presented on the screen, and increase or decrease the text size to his or her preferences.*

For every other modality of interaction, similar tests are presented to the user, and data about user impairments and preferences is collected. For example, the user is asked to perform different gestures, or asked to point to different locations on the screen to understand motor capabilities, to repeat out loud what he heard to understand hearing capabilities (figure 2 top), and asked to play memory and interpretation "games" with the goal of testing his or her cognitive capabilities (figure 2 bottom).


Figure 2: *UIA prototype. Examples of audio (top) and cognitive (bottom) tests presented to GUIDE users.*

From the results obtained in GUIDE user trials and from discussions with developers, we also know to be extremely important that UIA application must be presented to users in form of a simple and quick tutorial, so that elderly don't feel like they are being evaluated. If UIA takes too long, users will also lose interest, and will not want to use the system.

User information can be collected explicitly with the UIA, but also implicitly through run-time analysis of the user interaction logs. After the user has gone through all the UIA process, he or she starts interacting with different applications. Information concerning every task performed and modality used is saved by the system in logs. A rule-based inference motor will analyze this data and makes conclusions about user preferences and difficulties (for example, if the user makes consecutive errors when pointing to the screen for selection of a menu button, the system concludes he or she has difficulties using that modality and tries to increase the size of the buttons before suggesting a change in the modality of interaction). These conclusions enrich the data collected in the first process.

All data collected by the UIA and run-time processes are saved in a user model and used to adapt every application running on the framework [2].

**How to help users after a mistake has been identified?**

A supportive UI is one which tries to be aware at all times if a user is lost in the interaction, or if he or she is having too many interaction errors to be enjoying an efficient use of the application. Accordingly, one of the biggest challenges when guiding the user in the interaction, it's how to identify or perceive that the he or she is lost and when is the application or interaction generating errors. Only after identifying that, the application can then try to help the user and suggest alternative ways to achieve a desired goal. This is, however, a difficult task because at run-time a lot of dimensions are involved. If the user mistakes or misinterprets the interface structure and meaning, it can by itself result in interaction mistakes. There are also a lot of possible errors caused by changes in the context of the interaction, like the physical and social aspects of the environment. For example, if a user is interacting using speech input and the noise in the room increases, the system can fail to interpret the command issued because of the background noise, or a wrong command can be recognized instead (this can also happen when another person is speaking to the user at the same time of interaction).

Interaction mistakes will be identified in GUIDE by analyzing the interaction in run-time and by watching for unrecognized inputs. Because in this framework users can interact with UIs through different modalities (and devices), in a singular way or in a combined fashion, the system has to be alert for many different errors like:

- Unrecognized commands issued when speech input is performed.
- Selection of meaningless coordinates (coordinates not related with any UI interactive content) when pointing with finger.
- Unrecognized gestures performed by the user.
- Errors resulting from remote control commands.

- Repeated errors when interacting with each device or modality (consecutive errors could suggest a switching of modalities is required).
- Long periods with no selection registered but with screen navigation occurring (may suggest that the user is lost, or doesn't know what to do).
- Errors resulting from incomplete fusion of input modalities.
- Contradictory instructions from simultaneous input of different modalities.
- No input received after system started a task requiring user feedback.

Additionally, every time a change in context of interaction occurs, the system has to be alert for periods of inactivity or for unexpected inputs, and using the interaction logs the system tries to prevent some errors from happening when there is clear understanding of what are the causes.

A supportive UI has to be capable of helping the users every time there is a mistake in the interaction [4]. However, in modern applications help is a capacity "created ad-hoc" [4] meaning it was previously generated and it does not cover run-time situations not originally foreseen by the designers. For this reason, UI design does not cover every situation where a user needs help for responding to UI or interaction difficulties. Therefore helping the user is not something easy to do in a predefined manner before the user starts using the system, and requires some run-time "intelligence" from the supportive system or interface. For example, if a user is using speech input for menu navigation and his or her dog enters the room and starts barking, the system will receive a series of consecutive unrecognized inputs and the user will be in a situation that was not taken care off in the design process, which can result in aborting the interaction with the application.

As it is strongly based on multimodal interaction, one of GUIDE's ways of helping users after a mistake has been identified will rely on suggesting to the user a change in the modality of interaction. This change is however, based on each user preferences and characteristics firstly identified by the UIA and logs of interaction, as well as it is based in the context of interaction and task being performed at that moment[2]. So, as the user has already "ranked" modalities of interaction by preference (and based on constraints), every time an error results from repeated errors interacting with one single modality, another is suggested to the user, who accepts it (or rejects it) in order to continue the interaction. This will also be the procedure every time a change in the context of interaction happens [2] (for example, when the dog starts barking, the system won't recognize the barks as speech commands – rather, barks will be interpreted as background noise - and will suggest to the user continuing interacting using pointing).

Another way of helping users is to present to them relevant information related with the context of the error they have just made, like presenting alternative modalities of interaction and showing how to use them when a change in the context of interaction happens, showing information related with the task they are performing every time there are errors in the recognition of modalities or long pauses in the interaction (for example when the user is pointing and trying to select an area on the screen where there are no interactive UI items, show him or her where the buttons are by highlighting them). However, GUIDE main focus is helping users proceed with the interaction in an alternative way even when it's not possible to detect the cause of the error.

Finally, every time an error occurs, the Avatar engine will also be called for a more "personal" interaction between the system and the user (meaning, the Avatar presents the explanation of the error to the user, shows how changing modalities can solve the problem or just points the user to using an alternative modality when an error arises). In this way, it's almost like together they can find a solution to the problem or "find a way out" of the mistake.

**How to present content and interaction possibilities in the most suitable way to the users?**

The main problem with developing interfaces for elderly or disabled users is the great diversity existing in terms of user characteristics and user impairments. It is common for an elderly user to have more than one impairment (for example, poor hearing and poor vision), as it is usual to observe a lot of differences between each of these users. This means that what is good for one user can also, and at the same time, be inappropriate for several others. For example, an elderly user with hearing difficulties can interact with a visual interface without any problem, but one with severe visual impairments cannot, and need an interface with audio input and output for efficient interaction. However it is not expectable that developers will implement different versions of the same application, so the framework has to ensure the ways of interaction are adapted to the user characteristics.

GUIDE will offer elderly users adaptation mechanisms capable of adapting UI elements to each user characteristics. After the user has gone through the UIA and the system has collected enough information, the user is assigned to one UP [1]. Using the information about each user, GUIDE adapts each UI to fit the UP interaction patterns. This is only possible because GUIDE asks for extra information in each application development, so every UI is implemented using HTML, JavaScript, and CSS languages to what the developers add WAI-ARIA[ii] annotations providing semantic information about UI components. In this way, for every application, GUIDE will derive and keep an Application Model (AM), which is nothing more than an abstract interface that saves information about the structure of the UI and identifies each UI element present. This facilitates adaptation to different interaction contexts as well as to different types of users (users that belong to different UPs), because every time a user calls for an application, the system uses its application model and considering the interaction context

and user characteristics, modifies UI elements not appropriate for the user. For instance, when a user with visual impairments calls for an application formed by a visual menu and some text content, GUIDE consults its AM and "knowing" the user characteristics as well as "observing" no change in the interaction context, loads the UI increasing the size of the buttons originally defined and uses audio and visual output modalities.

In what concerns the developers control over this UI adaptation, GUIDE will adopt one of three adaptation schemes depending on the level of freedom given by the developer to change the application original properties (CSS and HTML): In "Augmentation", GUIDE won't be able to change any UI components, only making some overlay of output modalities (for example, adding audio output to a visual interface); in "Adjustment" GUIDE has permission to adjust UI component parameters as well as also making "augmentation" (for example, adding audio output to a visual interface and also changing UI colors for a higher-contrast); and finally in "Replacement" the developer gives total control to GUIDE, making possible the substitution of UI components as well as "augmentation" and "adjustment" (for example, adding or removing buttons, as well as adjusting colors and adding audio output to a visual interface).

Additionally, all interfaces must be capable of listening for user commands at any time of the interaction so that modifications to the interaction and presentation can be done at run-time, if the user is not satisfied with the current configuration. For example, if a user says "bigger buttons" or makes a gesture to increase the volume, the interface must adapt and reflect these changes (by reloading the UI or modifying output parameters).

## CONCLUSIONS

For the development of supportive multimodal user interfaces to be a reality, we have to make sure that the user's characteristics are known to the application. As well, the application has to be capable of instructing the users about all the ways of interacting with it, and make sure that adaption and UI help is presented to users in a personalized fashion. GUIDEs UIA, multimodal interaction and UI translation and adaption, were presented in this paper as possible solutions which can help in the deployment of supportive user applications without asking much more additional effort from the developers.

## REFERENCES


1. Biswas, P., Langdon, P.: Towards an inclusive world – a simulation tool to design interactive electronic systems for elderly and disabled users. Proc. SRII, 2011.
2. Coelho, J., Duarte, C.: The Contribution of Multimodal Adaptation Techniques to the GUIDE Interface. In: Stephanidis, C. (ed.): Universal Access in HCI, Part I, HCII 2011, LNCS 6765, pp. 337-346. Springer, Heidelberg (2011)
3. Garcia Frey, A., Calvary, G. and Dupuy-Chessa, S. Xplain: an editor for building self-explanatory user interfaces by model-driven engineering. In Proc. of the 2nd Int. Symp. on Engineering Interactive Computing Systems: EICS 2010, pp 41-46, ACM. Berlin, Germany, June 19-23, 2010.
4. Myers, B. A., Weitzman, A. J. Ko., and Chau, D. H.. Answering why and why not questions in user interfaces, in CHI'06: Proceedings of the SIGCHI conference on Human Factors in computing systems, pages 397-406, New York, NY, USA, 2006. ACM


[i] GUIDE– Gentle User Interfaces for Elderly People. http://www.guide-project.eu/.

[ii] http://www.w3.org/WAI/intro/aria