Managing Accessible User Interfaces of Multi-Vendor Components under the ULYSSES Framework for Interpersonal Communication Applications

Georgios Kouroupetroglou, Alexandros Pino and Constantinos Viglas

University of Athens, Department of Informatics and Telecommunications,
Panepistimioupolis, Ilissia, GR-15784, Athens, Greece, e-mail: koupe@di.uoa.gr

Abstract In this paper we present a novel approach, under the ULYSSES framework, to cope with accessible user interface management issues that arouse during the design, installation and maintenance of independently developed prefabricated software parts derived from multiple vendors and each one having its own GUI. ULYSSES constitutes a software engineering framework that facilitates Component Based Development for the domain of Computer Mediated Interpersonal Communication systems. Technical guidelines are provided for both component developers and system integrators.

1. Introduction

In the emerging convergence of voice and data communication technology, Computer Mediated Interpersonal Communication (CMIC) (or its equivalent term *e-communication*) launches an important societal role for all citizens. In e-communication either voice or text is commonly used to achieve synchronous or asynchronous (e.g. messaging, mailing) communication between two or more individuals. In some cases an alternative symbolic communication system (such as BLISS, PCS and REBUS) can be also utilized (Von Tetzchner, 1992). Of special interest is the case where in a communication session the two partners apply different meaning representation chosen among voice, text and symbols. Interpersonal communication is traditionally referred in the context of assistive technology. Recently however, general solutions have been proposed allowing communication between able-bodied and the disabled (Kouroupetroglou et al 1997, Viglas, Stamatis & Kouroupetroglou, 1998).

Universal Access (UA) according to Stephanidis et al (1998) entails the development of systems that can be used effectively, efficiently, and enjoyably by all users. Accessibility refers to two things (Benyon, Crerar & Wilkinson, 2001): a) physical access to equipment (in sufficient quantity, in appropriate places, at convenient times), and b) the operational suitability of both hardware and software for any potential user (even effortful participation qualifies as minimal accessibility). Attainment of UA is crucial for the design and the implementation of interactive CMIC systems (Emiliani et al 1996, Kouroupetroglou, et al, 1995). Consequently it is imperative that we architect CMIC applications for maximum adaptability to changing requirements for accommodating both the users' differences as well as the user characteristics changing with time.

Typically a Component Based Development (CBD) platform prescribes requirements that must be satisfied by component interfaces, and it provides facilities that support communication and coordination among those components (Allen, Garlan & Ivers, 1998). Furthermore, CBD has better suitability for a Design for All in HCI, compared to development methods closer to the physical level of interaction (e.g. presentation-based, physical task based and demonstration based) (Stephanidis et al 2001).

In this paper we present a novel approach, under the ULYSSES framework, to cope with user interface management issues that arouse during the design, installation and maintenance of pre-fabricated components derived from multiple vendors. ULYSSES constitutes a software engineering technical environment that facilitates CBD development for the domain of CMIC (Pino & Kouroupetroglou, 2000). According to ULYSSES, each component can be developed independently, complying with specific technical guidelines and requirements and incorporates its own specific Graphical User Interface (GUI). Thus the system integrator has to efficiently arrange, resize, fine-place and fixate the various GUIs of different components in order to constitute a single GUI for a specific product.

2. The ULYSSES framework

Component Based Development (CBD) consists of a recent software engineering approach that promises to fulfill the emerging need to produce flexible, scalable, adaptable and customizable cost effective applications on time. Components do not exist in isolation, but within an architecture that provides context across a technical domain or a

business. First steps were taken already in constructing internet-based CBD infrastructures that allow teams around the world to collaborate on every phase in the life cycle of a global software product (Gao, et al 1999). Componentization impacts the entire life cycle of an application, including the activities of analysis, design, acquisition, build, assembly, deployment, adaptation and replacement. A CBD framework is a software engineering technical environment facilitating the construction and delivering of domain specific applications out of independently-developed prefabricated parts. CBD frameworks are becoming popular (Kobryn, 2000), (D'Souza & Wills, 1999).

ULYSSES is a framework we developed (Pino & Kouroupetroglou, 2000), which offers a CBD infrastructure in the domain of CMIC. It is based on existing standards, domain engineering, looking for commonalties and variability of possible products covering a wide range of user groups including the disabled. Proprietary software architectures for CMIC systems have been already proposed (Kouroupetroglou et al, 1996, Viglas et al 1998).

ULYSSES components are likely to be self sufficient and independent of each other. They are polymorphic entities of different scale developed with different software technologies by independent vendors. ULYSSES adopts a three-layer architecture scheme: component (or construction), services and application (or integration). Components have their own intrinsic architecture and offer services that form the service architecture. The services can be reused as high-level concepts and integrated into the overall application architecture purely by knowing about the interface. The sequencing and use of the services can be easily modified. Likewise the use of the services is independent of the implementation and can be replaced with a new component that offers new capabilities or uses new technology.

CMIC applications under ULYSSES may incorporate various kinds of components: a) with their own visible GUI, like an on-screen keyboard or a message editor, b) with a speech user interface (SUI), like the output of a text-to-speech system, c) with a combination of GUI/SUI, and d) without any particular UI, like a natural language syntactic parser. The COM+ model and its event services have been adopted in ULYSSES as the mechanism for connecting components.

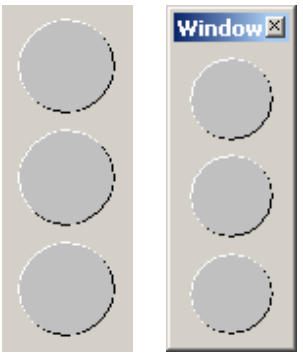
For component developers ULYSSES provides an engineering-for-reuse environment with guidelines and tools to build software components, which can operate effectively and interact with each other transparently, without even being aware of each other's existence. This facilitates the convenient production of reusable components for a family of products rather than single monolithic systems. Therefore, components with different functionality, developed by independent manufacturers and with various programming languages can be deployed. Furthermore, ULYSSES grants a process of engineering-with-reuse for system integrators to take advantage of the reusable assets produced during engineering-for-reuse. Integration standards in ULYSSES are typically specified using a combination of informal and semi-formal documentation. On the informal side guidelines and high level descriptions of usage patterns, tips and examples are included. On the semi-formal side it provides a description of an application programmers' interface (API) that explains what kind of services are provided by the infrastructure. APIs are formal to the extent that they provide precise descriptions of those services-usually as a set of signatures, annotated with pre- and post- conditions. ULYSSES supports four types of components: 1) specific to a CMIC product or family line, 2) CMIC domain specific components, and 3) domain-independent components (reusable across domains) and 4) technical infrastructure components (standalone components). The last two cases are also referred as components of an open market.

3. User Interface Management under ULYSSES

An integrator under ULYSSES has to provide a single UI of the final CMIC system by combining several independently-developed prefabricated components, each one having its own UI. An additional constrain is that components have been compiled separately, and the integrator has not the opportunity to change their code. To relief the end-user from the burden of arranging and dealing with multiple UIs, ULYSSES enacted guidelines to accomplish the following general specifications: a) *All Graphical User Interface (GUI) elements belonging to the CMIC application should be configured and arranged by the integrator after composing the application from multiple components. b) The application integrator must be able to manage the sizes and positions of all windows and GUI elements selecting to give the look of a single-window or a multiple-window application.*

Component **developers** should implement the above specifications complying with the following guidelines:

Each GUI of a component must have two function modes: the normal "run" mode (Figure1.a) and the "configuration" mode (Figure 1.b).

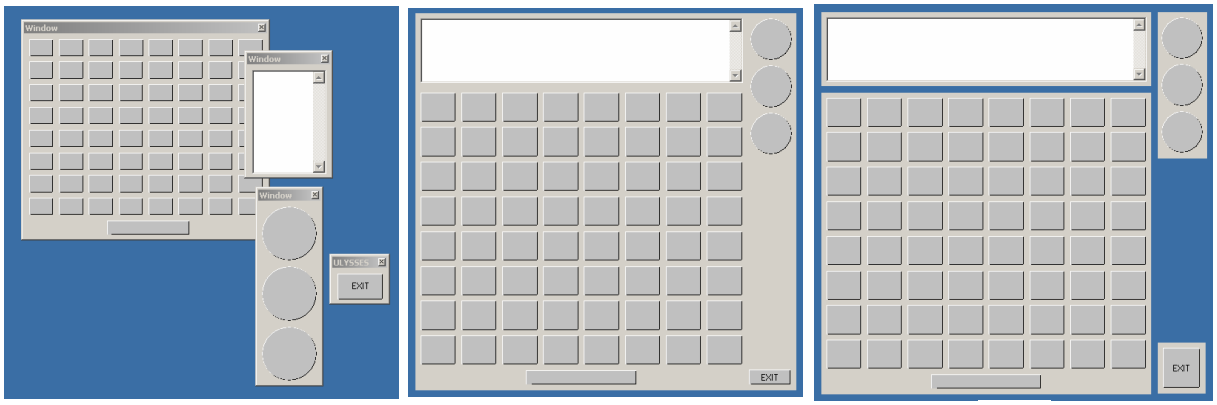


a) b)

Figure 1: Component's GUI in a) "run" and b) "configuration" mode.

- End users must have access and be aware only of the normal run mode in which window sizes and positions are fixed.
- End users cannot be able to resize, move or reposition any of the communication aid windows.
- When needed, the fact that the application consists of multiple windows must be completely transparent to end-users.
- Visible windows during the "normal run" mode must not have borders, control boxes, title bars and standard minimize, maximize and close buttons. Windows lacking of these characteristics, give the impression of a single window or user interface when they are placed close to each other with zero distance between them.
- The appearance of these windows must be centrally controlled by the operating system settings for foreground and background colors and font sizes.
- The configuration mode must be available only to system integrators and during this mode the windows must have "Sizeable ToolWindow" border, title bar and "close" button.
- During configuration mode the integrator must be able to resize and move the windows, while all windows elements and controls are resized and moved accordingly.
- Configuration mode must be activated with a preassigned hotkey and can be protected with a password known to integrators only.
- Pressing the "close" button during the configuration-mode will cause the application to return to the normal run mode and all the settings for window sizes and positions to be saved in configuration files or in the system registry in order to be available in future component activations.

Careful design of all user interface elements and some extra code is needed during the implementation. Each programming language may need different effort for realizing the required characteristics for the user interface. For example, in Visual Basic extra code and calculations are needed to make all window controls and elements resizable relatively to the parent window size, while in Java this can be done automatically. In Visual C++ a form can change its border from fixed to resizable ToolWindow at run-time, while in Visual Basic there is need for double forms (one fixed with no border and another resizable with ToolWindow border).



a) b) c)

Figure 2: Three phases of the UI during the integration process.

In the UI initialization phase of the CMIC application during the integration process, system **integrators** must manage several separate UIs belonging to multiple software components. At this phase all components have been installed and the application is ready to run. During the first application start-up, all components operate in normal

run mode and their UIs are shown on the integrator's screen using default size and positions according to the settings each developer used during the implementation of the component (Figure 2.a). There are specific steps that integrators must follow to set the application's final and unified user interface into an accessible and consistent format according to the available space on the screen and the number of visible windows:

1. Firstly, all windows must be set to window configuration mode. To do this the system integrator must give focus to the ULYSSES window and press a predefined hotkey. The ULYSSES window is a primitive GUI of the ULYSSES basic component. It consists of a simple window with only one button: "Exit". Pressing the hot key a message will be sent from the COM+ event system to all components with visible UIs causing them to enter the configuration mode.
2. All GUIs must be arranged on the screen in a way that can be visible and not overlapping or hiding each other.
3. The GUIs must be resized according to the user needs and the preferred layout on the screen. During this process the integrator must consider the consistence of font and control elements sizes between different components. Command buttons, for example, can have different sizes according to their functionality or the size of their parent window, but the integrator must ensure their homomorphy in order to avoid an odd and inconsistent appearance.
4. Positioning of the GUIs comes next. The communication aid integrator must place all visible UIs in suitable positions and in the right distances according to their functionality to achieve the final look preferred for the complete application. For example, if there are reasons for pointing out the difference between the various functional components of the application, this can be done by increasing the distances between their user interfaces (Figure 2.b). If there is need for the incorporated user interfaces to give the impression of a single unified user interface, they must be placed in a way that there is no visible space between them. At this point some fine placement and special attention may be needed to achieve with detail the final result. Keeping consistency and homomorphy in mind, the combination of all windows must present a robust user interface (Figure 2.c).
5. After the final GUI sizes and positions have been achieved, all settings must be saved and fixed. The integrator must press the "Close" button on the upper-right corner of each GUI in configuration mode. This will force all windows to turn to normal run mode and fix their sizes and positions, which will be "locked" for the end user. In all consequent executions of the CMIC application the multiple GUIs will retain their settings and only the integrator can reactivate the whole set up process.

Any specific accessibility option (such as scanning techniques) implemented either by the ULYSSES framework or by a dedicated component, according to the guidelines set by ULYSSES, will be inherited to the integrated application.

4. Conclusions

This paper has described an approach, under the ULYSSES CBD framework for the domain of CMIC, to cope with accessible user interface management issues that arouse during the design, installation and maintenance process.

As far as GUIs management is concerned, another known system is FRESCO, which is an OO user-interface system for developing windows-based applications supporting distributed component-based UIs (FRESCO, 1996). It includes, among others, a layout kit, which provides operations to manage the arrangements of GUI elements. The basic difference compared to the work described in this paper is that FRESCO is a CORBA based CBD environment for embedding and distributing graphic objects, while our approach is a COM+ framework facilitating the integration of independently developed prefabricated software parts derived from multiple vendors and each one having its own GUI.

Based on the guidelines and the tools of the ULYSSES framework, a number of user interfaces addressing both disabled and non-disabled users were built, under the AENEAS Project, on MS Windows 2000 platform, utilizing various accessibility options as well as different input/output devices and interaction elements. Multiple combinations of components implementing various functionalities and user interfaces for interpersonal communication applications only revealed the power and flexibility of the framework.

Acknowledgements

Part of the work reported in this paper was carried out within the framework of: a) the M-PIRO project (contract IST-1999-10982), funded by the IST Programme of the European Union and b) the AENEAS project (contract 98AMEA19), funded by the EPET II Programme of the Greek General Secretariat of Research and Technology.

References

Allen, R., Garlan, D. & Ivers, J. (1998). Formal Modeling and Analysis of the HLA Component Integration Standard. *Software Engineering Notes*, 23(6), 70-79.

Benyon, D., Crerar, A. & Wilkinson, S. (2001). Individual Differences and Inclusive Design. in Stephanides, C. (Ed) *User Interfaces for All*, (pp. 21-46). London: Lawrence Erlbaum Ass.

D'Souza, D. & Wills, A.C. (1999). *Objects, Components and Frameworks with UML: The Catalysis Approach. Addison-Wesley*, Reading, MA.

Emiliani, P.-L., Ekberg, J., Kouroupetroglou, G., Petrie, H. & Stephanidis, C. (1996). Development Platform for Unified Access to Enabling Environments, in Klaus, J., Auff, E., Kresmer, W. & Zagler, W. (Eds) *Interdisciplinary Aspects on Computers Helping People with Special Needs*, Oldenbourg, Munhen and Wien, Proc. of ICCHP' 96, July 17-19, Linz, pp. 69-75

FRESCO (1996) http://www.iuk.tu-harburg.de/Fresco/HomePage.html

Gao, J., Toyoshima, C.C. & Leung, D. (1999). Engineering on the Internet for Global Software Production, ACM Computer, 32(5), 38-47.

Kobryn, C. (2000). Modeling Components and Frameworks with UML. Communications of ACM, 43(10), 31-38.

Kouroupetroglou, G., Viglas, C., Stamatis, C. & Pentaris, F. (1997). Towards the Next Generation of Computer-based Interpersonal Communication Aids. in Advancement of Assistive Technology, Edit. G.Anogianakis, C. Buhler and M. Soede,. 3, IOS Press, p. 110-114, ????? 97, Proceedings of the 4th European Conference for the Advancement of Assistive Technology, 29 Sep. - 2 Oct. 1997, Porto Carras

Kouroupetroglou, G., Viglas, C., Anagnostopoulos, A., Stamatis, C. & Pentaris, F. (1996) A Novel Software Architecture for Computer-based Interpersonal Communication Aids. in Klaus, J., Auff, E., Kresmer, W. & Zagler, W. (Eds) *Interdisciplinary Aspects on Computers Helping People with Special Needs*, Oldenbourg, Munhen and Wien, Proc. of ICCHP'96, July 17-19, Linz, 715-720

Kouroupetroglou, G., Paramythis, A., Koumpis, A., Viglas, C., Anagnostopoulos, A. & Frangouli, H. (1995). Design of Interpersonal Communication Systems Based on a Unified User Interface Platform and a Modular Architecture, Proc. TIDE Workshop on User Interface Design for Communication Systems, Brussels, July 7, 1995, 8-17

Pino, A. & Kouroupetroglou, G. (2000). The ULYSSES Component Based Development Framework. Tech. Rep. 3.2. of AENEAS Project, Athens, Greece.

Stephanidis, C., Salvendy, G., Akoumnianakis, D., Bevan, N., Brewer, J., Emiliani, P.-L., Galetsas, A., Haataja. S., Iakovidis., I., Jacko, J., Jenkins, P., Karshmer, A., Korn, P., Marcus, A., Murphy, H., Stary, C., Vanderheiden, G., Weber, G. & Ziegler, J. (1998). Toward an information society for all: An international R&D agenda. International Journal of Human-Computer Interaction, 10(2), 107-134.

Stephanidis, C. & Kouroupetroglou, G. (1994). Human Machine Interface Technology and Interpersonal Communication Aids. Proc. of the Fifth COST 219 Conference: *Trends in Technologies for Disabled and Elderly People*, Tregastel, France: Gummerus Printing, 165-172.

Stephanidis, C., Akoumnianakis., D., Vernadakis, N., Emiliani, P-L., Vanderhheiden, G., Ekberg., J., Ziegler, J., Faehnrich, K.P., Galetsas, A., Haatajua, S., Iakovidis, I., Kamppainen, E., Jenkins, P., Korn, P. Maybury, M., Murphy, H. & Ueda.H. (2001). Industrial Policy Issues. In Stephanides, C. (Ed) *User Interfaces for All*, (pp. 589-608). London: Lawrence Erlbaum Ass.

Viglas, C., Stamatis, C. & Kouroupetroglou, G. (1998) Remote Assistive Interpersonal Communication Exploiting Component Based Development, Proceedings of the XV IFIP World Computer Congress, 31 August - 4 Sept. 1998, Vienna – Budapest, Congress: *Computers and Assistive Technology*, ICCHP'98, pp. 487-496.

Von Tetzchner, S.(1992). Use of Graphic Communication Systems in Telecommunication. In Von Tetzchner, S. (Ed.). *Issues in Telecommunication and Disability*. CEC, DG XIII, Luxembourg, 280-288.