Dialog Model Clustering for User Interface Adaptation

Guido Menkhaus and Sebastian Fischmeister

Software Research Lab, Department of Computer Science
University of Salzburg A-5020 Salzburg, Austria
{lastname}@SoftwareResearch.net

Abstract. In recent years, the dramatic growth of the PDA and mobile phone market demonstrates that users are willing to be constrained to small displays, limited storage and battery life, slow CPU speeds and data transfer, in the hope of achieving truly portable access to electronic data. Most of the limitations that users experience with current devices will disappear in future generations. These changes will not have an effect on the primary user interface constraint: the display size. The actual screen size will not change, since users demand devices that can be easily carried around and held in one hand. The article presents a hybrid approach to the generation of adaptive UIs based on a linking strategy of hierarchies of graphs. The nodes of this graph consist of clustered UI elements.

1 Introduction

From the increasing miniaturization of hardware arose a broad spectrum of mobile computers, which extend from mobile phones, computerized notepads, and PDAs to notebooks. This created a heterogeneity of computing platforms that stands in contrast to the past homogeneity of the desktop computer paradigm. This diversity often makes a transfer of knowledge difficult on how to build user interfaces (UI) between the platforms. The application of specific techniques for user interfaces, which are suitable for a desktop PC graphic user interface, to a miniaturized version fails. The objective is to develop UIs for the same application only once and not for each particular class of computing device.

There are research projects looking into new generation UIs that no longer consist of a conventional display. These UIs, although small in physical size, will no longer have size constraints concerning the UI. However, user acceptance seems to be low [1]. We think that traditional user interfaces still have a strong potential for improvement and that this technology will prevail in the near future on the consumer market [2].

The article presents an approach to UI adaptation. It is based on an abstract UI description, which is shared among different platforms. The adaptation technique tailors the UI description to minimize the mismatch between its presentation and the platform's capability to present it.

The remaining of the article is organized as follows: The following section presents a short overview of UI architecture. Section 3 introduces the concept of adaptation mechanisms. A special adaptation technique is discussed in Section 3.2 along with related work in Section 3.1. Results are presented in Section 4 and Section 5 closes the article with concluding remarks.

2 User Interface Architecture

Model-based UI software development has introduced concepts and techniques that assist in the process of UI development [3, 4]. These concepts give developers a better understanding of the field of UI design. There are three phases in the process of UI design [5]: The *semantic level design* describes the tasks users should be able to perform using the application for which the UI provides the interaction means. The *syntactic level design* describes the structure and the interaction behavior of the UI. The *lexical level design* consists of the detailed description of the visual part of the UI. The semantic design level consists of the task model. The dialog model presents the syntactic design level. The lexical design level describes the platform and the presentation model [6]:

1. **Task Model.** The task model is a formal description of the service the user accesses. It is hierarchically organized and contains information regarding the trigger of a task, its precondition, postcondition, and the action of the task itself.
2. **Dialog Model.** The dialog model describes the syntactic sequence of human-computer interaction through UI elements and determines the ordering of the set of tasks and actions. Since the task model is hierarchically organized, so is the dialog model. It is usually implemented as a sequence of windows consisting of a set of UI elements and a set of transitions that allows navigation from one window to the next.
3. **Presentation Model.** The presentation model accounts for the different devices from which a user may access a service. It maps conceptual elements of the dialog model onto platform specific elements.
4. **Platform Model.** The platform model contains information about the capabilities, restriction, and limitations of the target platform. This model is usually exploited dynamically at run-time.

The main obstacle to single authoring is the growing number of networking enabled devices with a wide variety of UI capability. One of the main differences they share is different screen size. How to enable content to be adapted to various screen sizes? The same content may require varying numbers of windows to display and a different navigational structure, depending on the platform. For example, content fitting on one PC window may require 3 windows on a mobile phone. Yet, all these windows originate from the same, single authored UI. The platform model delivers information about the limitations and restrictions of the target platform. The adaptation process exploits this information for adapting the content and the navigational structure.

The challenge is to remodel the UI elements into a composition which allows the user to adequately interact with the application and that respects the dialog model independent of the target device.

2.1 Short Overview of Adaptation Mechanisms

Until recently, UI software has been designed with a specific environment in mind, resulting in ideal properties in this environment. Any other environment entails an adaptation of the system resulting in an output having properties less optimal than the ideal properties. An ideal adaptable software maintains the same ideal properties under all environments. However, in practice it is impossible to maintain the ideal properties under a varying environment. Therefore, the goal of the adaptable software system is to result in an output having properties close to the ideal properties. Following the notation of [7]: Let s be a software system. E is the space of environments hosting all possible environments. $e_i \in E$ is one environment having a set of properties $p_i \in P$. Let $I \subset E$ be the input space, and $i_i \subset e_i$ provides the inputs of the environment e_i. $O \subset E$ is the output space, and $o_i \subset e_i$ the output space of the system s under the environment e_i. A computation of the system s produces the results o_i with properties p_i in an environment e_i: $s(i_i) \rightarrow o_i : o_i \models p_i, p_i \in P, i_i, o_i \in e_i$ The equation states that s guarantees the properties p_i under the environment e_i. An ideal adaptable system maintains the same properties under all environments. Since there is no such ideal system, the objective of the adaptable software system is to aim at an output with properties close to the ideal properties. The adaptation process adaptation can be regarded as an energy minimization problem, with the energy function:

$$E(e_i) = \frac{1}{n} \sum_{j=1}^{n} d(p_{i,j}, p_{i,ideal})^2$$

with $p_{i,j} \in p_i,\ j = 1,\ \ldots, n$ being specific properties of the environment e_i and $d(\cdot, \cdot)$ being a distance measure between properties. The energy is minimal, when the ideal properties are met after the process of adaptation.

3 Dialog Model Adaptation

In this article a dialog of the dialog model will be represented by a two-dimensional discrete function $UIelement(x, y)$, which is digitized both in spatial coordinates and feature value: $dialog = [UIelement(x, y)]_{P \times Q}$ where $P \times Q$ is the size of the dialog, (x, y) denotes the spatial coordinate and $UIelement(x, y) \in UIE$ the type of UI element from the set of available abstract UI elements UIE. Without loss of generality we consider only the case, where $Q = 1$.

Grouping UI elements that implement the dialog model into a hierarchical structure of windows is the essential step our adaptation process that leads to single authoring. For this, the dialog model adaptation process partitions the UI elements implementing the dialog model into non-intersecting regions, such that each region satisfies a homogeneity predicate and the resulting hierarchical structure of windows minimizes an energy function.

Formally, the process of adaptation of the dialog model can be defined as follows: If a dialog model consists of a set of $UI\ elements$ and P is a homogeneity predicate, then the adaptation of the dialog model is a partitioning of UI elements into a set of connected regions $(r_1, r_2, \ldots, r_n)$, which will eventually be converted into a hierarchical navigationable structure of windows, such that:

$$
\begin{aligned}
dialog &= \cup_{i=1}^{n}(r_i \backslash navigationUIelements(r_i)) \\
r_i \cap r_j &= \emptyset, i \neq j \\
& r_i \text{ is a connected set of regions} \\
P(r_i) &= \text{true}, i = 1, \ldots, n \\
P(r_i \cup r_j) &= \text{false, if } r_i \text{ is adjacent to } r_j
\end{aligned}
$$

The adaptation of the dialog model partitions a dialog into regions of non-intersecting UI elements complying with a homogeneity predicate. A user accessing a service supported by a dialog model needs to navigate from one region to the next region. However, not all navigation elements are in the original dialog. Thus, they have to be integrated into the regions, resulting from the adaptation process. The set of all UI elements in the regions equals the UI elements in the original dialog plus the integrated new UI elements dedicated to the navigation between the regions, the $navigationUIelements(r_i)$.

3.1 Related Work

The above definition of the process of adaptation is very similar to image segmentation as defined in [8]. Analogous to segmentation and clustering processes, the more context and domain information is known beforehand and integrated into the process, the better the process' results. Approaches exploring dialog model adaptation can broadly be divided into two categories. Processes of the first category do not consider context knowledge such as screen size during design time. They work bottom-up and rely uniquely on dynamic adaptation of the dialog model. The other category explicitly uses top-level domain and task model knowledge during design time. The processes are configured with a priori known target contexts.

Adaptation without Context Knowledge. One of the key features of the Microsoft Mobile Internet Toolkit (MMIT) is the ability to transcode content of a Web form to the screen size of the target device. This process is called paginating and chunking [9]. Pagination is similar to the fragmentation process in IBM's Websphere [10]. The places where the process of pagination cuts a Web forms into two different pages depends on the device screen's size requirements, but not on semantic dependencies between controls. For example, textual information explaining the use of a button and the button itself are modeled as two distinct controls. If these two UI elements were grouped into two different regions after the adaptation process, the adapted dialog model had low usability. In MMIT, this process may be controlled by inserting *Panel* controls, which avoid page breaks between controls. The result is a set of regions, which can be navigated in a linear, sequential way only. To access the last control of a dialog, each new window (region) has to be linearly traversed.

Adaptation with Context Knowledge. Approaches of this category allow the designer to submit to the adaptation process specific configuration information concerning possible target device types [11]. The designer creates the appropriate configuration for each class of device. On the one hand, these configurations are best adapted to the requirements of the underlying service. On the other hand, a single modification of the requirements of the service entails the modification of each configuration.

The quality of the latter approach depends on the configuration and the type of content that is presented. The first approach has the drawback of working only on syntactic information. We propose a hybrid approach that combines the advantages of the first bottom-up working approach (fast design, non need to produce sophistic configuration data) and the latter top-down approach (integration of semantic information).

3.2 Dialog Model Adaptation using UI Element Clustering

The two main challenges of the hybrid approach to dialog model adaptation are:

1. How to incorporate low-level semantic information into the dialog model?
2. How to adapt the dialog model respecting the semantic information?

No current approach to UI software development implements a clear separation of presentation, dialog and task model, nor does any current markup language supports the feature of integrating adaptation configuration that could guide any adaptation process. We have developed the Multi UI Single Application (MUSA) model and Event Handler Graph XML. MUSA implements a clear separation of task, dialog, presentation and platform model. The Event Graph Handler XML representing the dialog model is a markup language that allows adding information to each element stating the semantic relation to its neighboring elements. For more information on MUSA and Event Handler Graph XML, refer to [12, 13]. In the next section, we introduce the dialog model adaptation process.

Dialog Model Clustering. The adaptation technique is based on a linking strategy of two hierarchies of graphs [14, 15]. The first hierarchy of graphs forms a syntactic based and static structure that guarantees that the resulting regions are connected. The second hierarchy is dynamically built up respecting the low-level semantic information integrated into the dialog model at design time. The two hierarchies of graphs implement the dialog model adaptation process.

The elements of the dialog model are placed as abstract UI elements into a stack of regular grids, as illustrated in Figure 1. In the lowest level of the stack, each cell of the grid corresponds to a single UI element. Each cell of level $i + 1$ represents a group of cells of level i. The adaption algorithm always forms linear structures of 3×1 cells. The cells overlap in such a way that the outer cells on level i belong to two cells of level $i+1$. The cells in a group of level i, represented by a cell of level $i + 1$, are called the subcells or the children of this cell. The

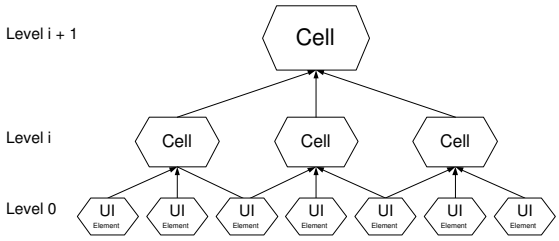

Fig. 1. Stack of a regular grid of cells that places a structure on a set of UI elements. Three UI elements form a cell on the lowest level. Cells on a lower level are candidates for cells on a higher level.

representing cell is called the parent of its children. The clustering of a set of UI elements into a set of regions is done within the boundaries of the induced stack of cells and is of primary interest. To come to the final set of regions, we dynamically build up a stack of regions. A UI element corresponds to a region on the lowest level. Adaptation of a window is performed by clustering regions of level i into regions of level $i+1$. However, regions can only be grouped within the boundaries of a cell in which they reside, as illustrated in Figure 2, and if they satisfy the homogeneity predicate. This guarantees that we cluster only connected regions.

A hierarchy of regions is built up by applying a clustering process to each cell while moving up the stack of cells. The clustering process stops at the boundary of each cell and the cell overlap is responsible some regions taking part in two cell-based clustering processes. The receptive field RF [15] of a region $r \in R_i$, with R_i being the set of regions of level i, is defined as the set of all regions on the lowest level, which represent the region r. The semantic information σ integrated into the dialog model is assigned to each region r. $\sigma(r)$ is the average value of the semantic information of each region's receptive field.

The adaptation process adapts dynamically the dialog model by composing and decomposing UI elements of the dialog model into a set of regions, which results finally into a hierarchical structure of linked windows. The process consists of the following four phases:

- **Bottom-up Clustering.** Regions of level i are grouped into regions of level $i+1$ within the boundaries of their cell and satisfying a predicate P.
- **Top-down Separating.** Regions that fail to group on level i are separated recursively down to level 0.
- **Horizontal Separation.** Large-sized regions of level i, especially when they contain a single UI element, are split into smaller regions of level i.
- **Relinking.** The user should be able to navigate from one region to the next region. To ensure usability, regions are relinked by integrating additional navigation UI element.

Bottom-up Clustering. The clustering process determines the set of connected regions of level i of a specific cell and groups them. In order to form a new region

r_{i+1} (the subscript indicates the level) in a cell c_{i+2}, the set of subcells of c_{i+2} are determined. Each subcell has a set of regions associated that are candidates for clustering into r_{i+1}. A region s_i groups into the region r_{i+1}, if it satisfies the homogeneity predicate $P(r_{i+1} \cup s_i) = \text{true}$.

The clustering process is illustrated in Figure 2. Two regions s_i, t_i are connected i.e., they have a common subregion u_{i-1} and will be grouped into the region r_{i+1}. Regions of the lowest level are connected with their neighboring regions. The overlapping structure of the stack of cells guarantees that the grouping process considers only those regions, which are connected or have a path of connected regions on the lowest level, the UI element level. The homogeneity

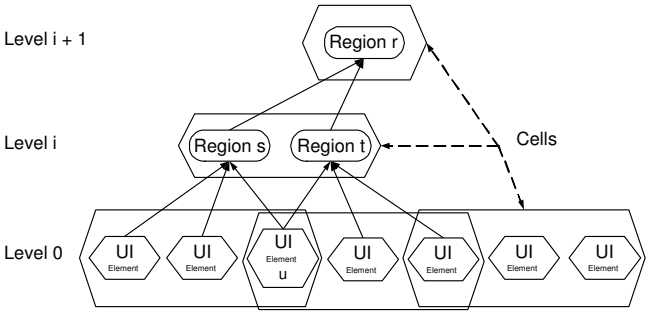

Fig. 2. Grouping process. Regions are grouped within the boundary of a cell.

predicate decides if regions are clustered or not. The predicate consists of two parts, which both need to evaluate to true; $P(r) = Size(r) \wedge Context(r), r \in R_i$.

- **Size.** On different platforms a dialog is displayed with a varying number and size of UI elements. If the size of a region and its parent region is lower than a predefined threshold (e.g., three times of the screen size) the regions are clustered, otherwise they are separated, either horizontally or top-down. The size of a region is the size of its receptive field.
- **Context.** The designer of the original dialog model integrates in it semantic information. The information deals with the semantic relation of a UI element with its neighboring UI elements. A region s_i and its tentative parent region r_{i+1} are grouped if their semantic intent does not exceed a predefined threshold $d(\sigma(s_i), \sigma(r_{i+1})) < \Theta$. In the current version of the adaptation process, we simply assign integer values to UI elements, to indicate semantic similarity. $d(\cdot, \cdot)$ is a distance measure like the Euclidian distance.

Top-down Separating. If the grouping process fails, because a region s_i does not satisfy the homogeneity predicate P, the region need to be separated from its connected region t_i. The region need to be separated since they have a common subregion u_{i-1}, which needs to be assigned to a single parent region (Figure 2).

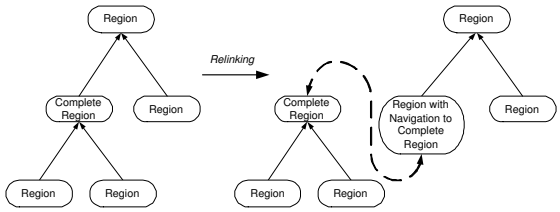

Fig. 3. A region containing a single navigation UI element will replace a *complete* region. The new region takes part in the building phase on behalf of the *complete* region.

The separation process assigns the common subregion to the region, whose semantic value is the most similar. This means that u_{i-1} is removed from the set of subregions of the other region. The process is recursively applied down to the lowest level. For level $i-1$ in Figure 2 it would be applied to u_{i-1}, the common subregion of s_i and t_i, and to those subregions of regions of level i, which have a common subregion with u_{i-1}.

The process of bottom-up clustering and top-down separating can be seen as an energy minimization problem. The energy is defined as follows:

$$E = \sum_{i,r_j \in R_i} \frac{\sum_{u_{j-1} \in \kappa(r_j)} \left\| RF(u_{j-1}) \right\| d(\sigma(u_{j-1}), \sigma(r_j))^2}{\sum_{u_{j-1} \in \kappa(r_j)} \left\| RF(u_{i-1}) \right\|}$$

with κ being the mapping that assigns to a region r its subregions. Two regions are top-down separated such that the energy of the resulting regions in the hierarchy of graphs is minimal with respect to the size requirements of the homogeneity predicate.

Horizontal Separation. If the size of a region r_i prevents it from clustering with other regions although it could from the context part of the homogeneity predicate's point of view, it is split into a sequence of n smaller, mutually linked regions $r_{0,i}, r_{1,i}, \ldots, r_{n,i}$. E.g., a lengthy text message is split into a sequence of regions containing each a part of the text message. Only the head of the sequence continues to take part in the grouping process.

Relinking. A region that cannot further be clustered with other regions into a region of a higher level is called *complete*. A *complete* region that has reached the threshold of maximal allowed size or that cannot further be clustered from a semantic context point of view does not drop out of the grouping process. Instead, a new region is created containing a single navigation UI element pointing to the *complete* region. The new region takes the place of the *complete* region and continues the grouping process on behalf of it. The process is illustrated in Figure 3. The set of regions resulting from the adaptation process are transformed applying the presentation model into a hierarchical structure of windows containing concrete UI elements.

4 Results

To illustrate the adaptation technique of a dialog model we have implemented a location-based message board [12]. The message board contains location specific information and users can read and store messages on the message board. A mobile user moving from location to location accesses different message boards depending on the geographical position. Different users use different devices to access the message board such as laptops, PDAs, or mobile phones. The dialog model that results in the graphical UI on a HTML browser is shown in Figure 4(a). The same dialog model but this time adapted to the small screen of a mobile phone is shown in Figure 4(b). There are two things to note. Firstly,

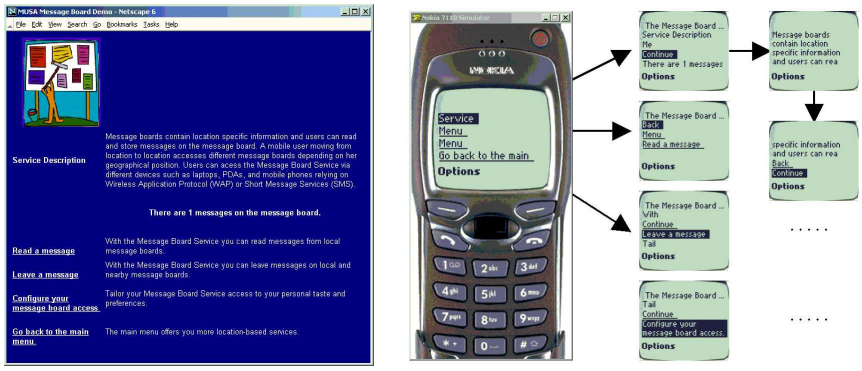

Fig. 4. (a) HTML browser showing the Message Board "Main Menu". (b) WML browser showing the Message Board "Main Menu".

the menu is hierarchically structured into a two level menu, with a main menu containing links to each menu item, which are presented on their distinct screen. The main menu is created during the relinking process of the adaptation and is not present in the original dialog model. The grouping process groups the newly created navigation UI elements together, which results in the main menu. Second, the service descriptions, which are lengthy text messages, are split into a series of screens, which are linked with each other. The user navigates with the "Continue" and "Back" links from one screen containing part of the description to the next screen.

5 Concluding Remarks

The article has presented a new approach to dynamic UI adaptation. The adaptation process is based on bottom-up clustering and top-down separation using

low-level semantic context information. It results in a hierarchical structure of windows by clustering, separating, and relinking regions. The process is guided by low-level semantic information that is provided by the designer of the dialog model at design time. The adaptation process remodels dynamically a presentation of the dialog model to better fit it to the current platform model.

The presented experiments with the dialog model adaptation technique are promising and show that the concept is sound. The use of the hierarchy of graph has been proven flexible and is a viable concept for future UI development.

In our future work, we will elaborate the adaption algorithm to include user specific settings such as window size of the running application or user-preferred font size.

References

1. Alpert, M.: Machine Chic. Sci.Am (2002)
2. Marcus, A., Chen, E.: Designing the PDA of the Future. Interactions **9** (2002) 34–44
3. Pinheiro da Silva, P.: User Interface Declarative Models and Development Environments: A Survey. In Palanque, P., Paternò, F., eds.: Proceedings of DSV-IS2000. Volume 1946 of LNCS., Ireland, Springer-Verlag (2000) 207–226
4. Szekely, P.: Retrospective and Challenges for Model-Based Interface Development. In Bodart, F., Vanderdonckt, J., eds.: Design, Specification and Verification of Interactive Systems '96, Wien, Springer-Verlag (1996) 1–27
5. Schlungbaum, E.: Model-based User Interface Software Tools - Current state of declarative models. Technical Report 96-30, Graphics, Visualization and Usability Center, Georgia Institute of Technology, Atlanta (1996)
6. Puerta, A.: A Model-Based Interface Development Environment. IEEE Software **14** (1997) 41–47
7. Seng, A.R.: An Adaptability Framework for Mobile Applications (1999) `http://citeseer.nj.nec.com/290640.html`.
8. R.Pal, N., K.Pal, S.: A Review on Image Segmentation Techniques. Pattern Recognition **26** (1993) 1277–1294
9. Microsoft: Mobile Internet Toolkit - QuickStart Tutorial (2002)
10. Britton, K., R.Case, A.Citron, Floyed, R., Li, Y., Seekamp, C., Topol, B., Tracey, K.: Transcoding. Extending e-business to new environments. IBM Systems Journal **40** (2001) 153–178
11. Mandyam, S., Vedati, K., Kuo, C., Wang, W.: User Interface Adaptations: Indispensable for Single Authoring. In: W3C Workshop on Device Independent Authoring Techniques, SAP University, Germany, W3C (2002)
12. Fischmeister, S., Menkhaus, G., Pree, W.: MUSA-Shadow: A Location-Based Service Supporting Multiple Devices. In: Proceedings of Pacific TOOLS, Sydney, Australia (2002) 71–79
13. Menkhaus, G.: An Architecture for Supporting Multi-Device, Client-Adaptive Services. Special Volume of the Annals of Software Engineering Journal on OO Web-based Software Engineering (2002)
14. Nacken, P.: Image Segmentation By Connectivity Preserving Relinking in Hierarchical Graph Structures. Pattern Recognition **28** (1995) 907–920
15. Hartmann, G.: Recognition of Hierarchically Encoded Images by Technical and Biological Systems. Biological Cybernetics **57** (1987) 73–84