

HCI Patterns as a Means to Transform Interactive User Interfaces to Diverse Contexts of Use

Jürgen Engel¹, Christian Martin¹, and Peter Forbrig²

¹ Augsburg University of Applied Sciences, Faculty of Computer Science,
Friedberger Strasse 2a, 86161 Augsburg, Germany

² University of Rostock, Institute of Computer Science,
Albert-Einstein-Strasse 21, 18059 Rostock, Germany
{Juergen.Engel, Christian.Maertin}@hs-augsburg.de,
{Peter.Forbrig}@uni-rostock.de

Abstract. This paper introduces a pattern-based method for transformation of user interfaces of interactive applications to diverse contexts of use. The method is demonstrated with the help of various examples taken from existing software solutions. The related pattern transformation rules are derived from the samples and in turn expressed in a pattern format, the so-called transformation patterns.

Keywords: HCI patterns, user interface, pattern-based context transformation, transformation patterns, interactive systems, PLML.

1 Introduction

In the past electronic appliances and devices, e.g. telecommunication devices, usually were comprised of proprietary hardware. Nowadays such products are increasingly implemented on a software basis, e.g. as embedded applications, web clients or Apps. These types of interactive software systems steadily become more important.

As in most application domains users can choose from a vast variety of competing products usability and user interface aspects become substantial. But user interface development in particular absorbs a significant portion of the total development effort [15]. Emerging platforms, such as smart phones, other mobile devices, and new web technologies are the driving forces for even higher UI complexity.

Due to these facts we focus our research on options for automation in the UI development process. Here a major aspect is pattern- and model-based UI modeling and UI code generation. Within this paper we would like to share the findings of our ongoing work on transforming user interfaces to diverse contexts of use.

2 Related Work

Patterns and pattern languages have a growing impact on the disciplines of HCI and web engineering. Christopher Alexander introduced patterns for solving problems in architecture and urban planning. He defined a pattern as a three-part rule which expresses a relation between a certain context, a problem, and a solution [1]. In the

1990s patterns were adopted by software engineers and software architects for promoting the reuse of high-quality design solutions [8]. For over a decade now patterns and pattern languages have also entered the fields of HCI, usability engineering [12], user experience [18] and organizational workflow [9].

Over the years various HCI pattern catalogues have been developed and published providing valuable and reusable design knowledge. Examples are Jenifer Tidwell's *Designing Interfaces*, *Patterns for Effective Interaction Design* [17], Martijn van Welie's *Patterns in Interaction Design* [19], Todd Coram's and Jim Lee's *A Pattern Language for User Interface Design* [3], or the community-driven UX and UI pattern library *Quince* operated by Infragistics [10].

However, most of the available pattern collections lack an appropriate organizational structure in order to facilitate pattern selection and ensure the overall coverage of domain dependent and independent modeling and design problems. Manageability aspects of various existing UI pattern catalogues are discussed and compared in [4]. In [13] a structured approach both for designing hierarchically organized HCI pattern languages and controlling the selection of the really needed patterns during the software development process is introduced.

Another significant shortcoming in the area of patterns is that the various authors usually describe their patterns in different and inconsistent styles. This makes it hard or even impossible to search, select and reference patterns across pattern collections. In a workshop which has been held during the CHI 2003 conference the participants aimed for unification of pattern descriptions and guidance for the authors. Hence the Pattern Language Markup Language (PLML) has been constituted. The current version PLML v1.1 stipulates that the documentation of a certain pattern should consist of the following elements: a pattern identifier, name, alias, illustration, descriptions of the respective problem, context and solution, forces, synopsis, diagram, evidence, confidence, literature, implementation, related patterns, pattern links, and management information [7].

The use of patterns can support software engineering activities in all phases of the software development life-cycle. A comprehensive process for user interface engineering using patterns is e.g. introduced in [16].

In [11] it is demonstrated that HCI patterns can be used to migrate existing user interfaces of web applications across platforms considering the screen size of the respective user devices and the complexity of the data architecture. Two different approaches can be used for this type of migration: redesign and reengineering. While redesign is a simplified procedure for migrating directly from one to another platform-specific user interface, reengineering consists of an intermediate step of creating an abstract platform-independent UI model. In a case study redesign with navigation patterns is illustrated.

The current state-of-the-art in the growing field of HCI patterns and pattern-based software development approaches is documented in [2].

3 Pattern-Based Context Transformation

In [5] we introduced a pattern- and model-based framework for partially automated generation of UI code. One of its key components is a pattern repository containing

the pattern definitions according to PLML v1.1 [7]. The patterns are hierarchically organized according to their respective levels of abstraction as outlined in [13]. On a top level view, the related taxonomy distinguishes patterns for navigation, content presentation and user interaction issues. The framework supports its users designing an abstract user interface model which serves as the basis for transformation into context-specific semi-abstract UI models. Finally the code of the target user interface is automatically generated from these semi-abstract models.

The UI model construction and transformation processes are based on diverse models, including task model, user model, device model, environment model, and data architecture model.

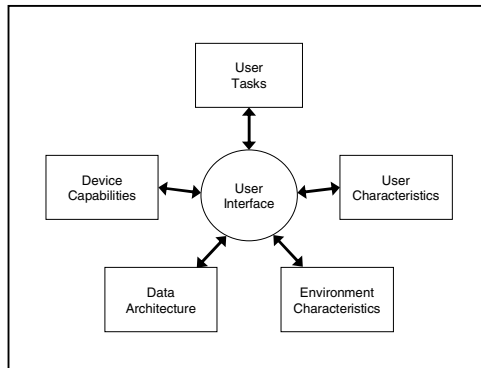


Fig. 1. User interface influence factors

The task model commonly incorporates the user activities, their types, hierarchical structure and temporal relationships. Preferably task models are represented as Concur Task Trees (CTT) [15]. In the user model we hold information about the user's preferences, expertise, possible disabilities and grade of distraction, for instance when the user is in charge of driving a car while interacting with a mobile device, e.g. a phone or navigation system. The device model holds data about input and output capabilities, screen size and resolution, bandwidth, storage capacity, processor power, and available widget sets. If not explicitly modeled, we use some of these values to determine the device's media capabilities, such as text, graphics, sounds, and audio and video streaming. The environment model may indicate possible problems due to light irradiation, noise, or pollutant. Especially for web applications the data architecture model contains information about the complexity of the relevant content to be provided to the user [6], [11]. From these models one can construct various contexts of use during design time.

The starting point of the pattern-based UI model transformation process is a set of individual HCI patterns. Table 1 provides an overview of the patterns we had in focus during our analysis work even though the list is not depletive. For lack of space it is not possible to provide here the full-blown pattern definitions according to PLML

Table 1. Alphabetical list of HCI patterns used for Transformations

ID	HCI Pattern Name	Brief Description
01	Accordion	Stack panels vertically or horizontally and open up one panel at the time while collapsing the other panels [19]
02	Alphabetical Index	Index contains links to pages providing details of the indexed terms
03	Bread Crumbs	Hierarchical navigation path from top level to current page making each step clickable [19]
04	Center Stage	Put the most important part into the largest subsection of the page and cluster secondary content around it [17]
05	Collapsible Panels	Create panels that can be opened or closed independently of each other [19]
06	Contextual Horizontal Menu	A horizontal menu displayed temporarily depending on a certain context
07	Contextual Vertical Menu	A vertical menu displayed temporarily depending on a certain context
08	Directory Navigation	Grouping links for items of the second hierarchy level under headings of the first level [19]
09	Drop-down Menu	A list of items appears when invoking the menu and one of the displayed items can be selected
10	Fly-out Menu	Combination of horizontal menu with sub-menu flying out while hovering over the main menu-item [19]
11	Portal Site	One overall home-page leading to several sub-sites [19]
12	Keyword Search	Offer a simple search capability to the user for locating required content by specifying keywords
13	List Builder	Present the total list and provide editing functionality next to it [19]
14	List View	Show a simple list of items
15	Most Relevant First	Put the most important information in the first place of a container element, e.g. a list of items
16	One-Window Drilldown	Show each page within a single window and as a user drills down replace the window contents completely with the new page [17]
17	Permanent Horizontal Menu	An always visible single-row menu bar providing a consistent set of links leading to key sections
18	Permanent Vertical Menu	An always visible vertical menu providing a consistent navigation tool across all pages
19	Repeated Menu	Repeat the main navigation on the bottom of the page in order to minimize scrolling effort [19]
20	Tab Menu	Put groups of content onto separate panels which can be accessed via tabs
21	Teaser Menu	Show a partial menu with a capability to expand to the full-blown menu, e.g. using a “show more” link [19]
22	Tiled Sections	Define separate sections of content and lay them all out on the page together [17]
23	Top Link	Provide a link to the top of the page at locations in the main content [19]
24	Two-Panel Selector	Put two panels side by side – one displaying a set of selectable items and the other showing the details [17]
25	Wizard	Lead the user through the interface step by step, doing tasks in a prescribed order [17]

v1.1, but the pattern names and a brief description. For these patterns we have defined reasonable transformation directives for specific contexts of use. These rules in turn can themselves be regarded as type of transformation or mapping patterns. They consist of descriptions of a problem, i.e. the transformation problem, a context, i.e. the modeled contexts of use, and a solution, i.e. the transformation directive. Thus we have a unified method to describe both, the underlying HCI patterns and the transformation rules.

The pattern-based UI model transformation is depicted with the help of the following examples taken from the analysis of existing software solutions. Here it is intended to provide user interfaces of an individual application for different devices, i.e. desktop PC and smart phone with a limited screen size.

Figure 2 shows in its upper part the desktop PC version of the world-wide-web homepage of the German public TV station ARD [20]. For instance, the main navigation of the web page is implemented according to the “Permanent Horizontal Menu” pattern (pattern ID 17 in table 1) and consists of 12 menu entries. For the small device version of the UI this pattern is initially transformed to the same pattern, but the resulting horizontal menu bar is truncated to only one single menu item “Navigation” as shown within the left screenshot in the lower area of figure 2. When clicking on this menu item a new screen is opened up according to the “One Window Drilldown” pattern (ID 16), showing the original full amount of menu options, but now according to the “Permanent Vertical Menu” pattern (ID 18).



Fig. 2. Desktop PC and Mobile Version of the ARD Internet Homepage

There exist different types of pattern-based transformations. At first there is the option to use the same pattern in the source as well as in the target user interface, i.e. the sameness on the abstract pattern-level. A second possibility is to apply the same patterns, but to vary in extensiveness, e.g. to include more or less items in a menu bar or to incorporate much or sparse content into a screen. The third option is to completely reorganize the structure and replace patterns by one or multiple different

ones. Basically multiple transformation patterns for a specific pattern and a specific context may coexist, i.e. that there are transformation alternatives.

On basis this circumstance it is possible to derive a first transformation pattern definition as presented in table 2. Again, for lack of space, we merely included the transformation pattern attributes ID, problem, context, solution and illustration.

Table 2. Definition of Transformation Pattern #1

Attribute	Description
ID	T-001
Problem	How to transform a “Permanent Horizontal Menu” pattern (ID 17) under the given context?
Context	• Source: desktop PC user interface • Display size: 15 in. or bigger • Display resolution: 1024x768 or better • Target: smartphone device UI • Display size: 35x41 mm or alike • Display resolution: 176x208 px or alike
Solution	1. Apply a “Permanent Horizontal Menu” pattern (ID 17), but drastically reduce the amount of items, e.g. only one item named “navigation” 2. Apply a “One-Window Drilldown” patter (ID 16) in order to open up a new screen 3. Apply a “Permanent Vertical Menu” pattern (ID 18) including the original list of items.
Illustration	Please refer to figure 2.

The pattern-based transformation is not limited to navigation patterns, but can also be applied for content presentation and interaction patterns. Figure 3 shows in its upper part the main page of the internet email service of t-online.de, an online service of the German telecommunications carrier Deutsche Telekom AG [21]. This page is implemented according to the “Portal Site” pattern (ID 11). Amongst others it contains a “Two-panel Selector” pattern (ID 24) as indicated by the two ellipses in the screenshot. The left part allows for selecting an individual type of email messages, while the other provides a list of emails of the chosen type.

The fundamental email type selection function is transformed to a “List View” pattern (ID 14) implemented as entry page for the mobile UI. When clicking on one of the displayed list items a subsequent page is opened up according to the “One-Window Drilldown” pattern (ID 16) showing the realization of a “List Builder” pattern (ID 13) providing the list of respective emails. The particular list entries are arranged as multi-line items so that the most relevant information can be displayed on a small device. In summary the “Two-panel Selector” pattern of the large screen is replaced by several different patterns within the mobile UI.

The related transformation pattern definition reads as shown in table 3.

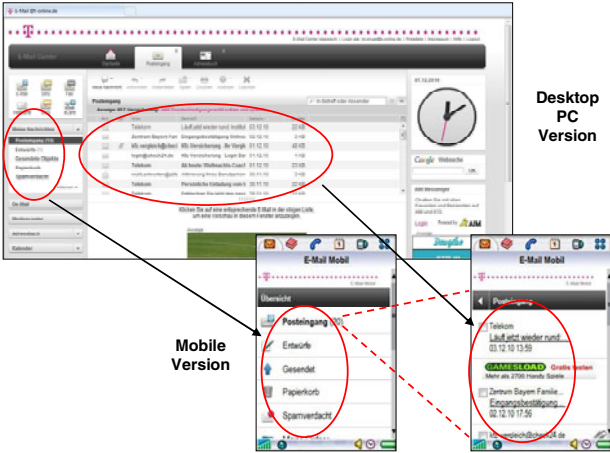


Fig. 3. Desktop PC and Mobile Version of the Email Service t-online.de

Table 3. Definition of Transformation Pattern #2

Attribute	Description
ID	T-002
Problem	How to transform a “Two-Panel Selector” pattern (ID 24) under the given context?
Context	Transformation from desktop PC to smartphone UI as contemplated in pattern T-001
Solution	1. Apply a “List View” pattern (ID 14) incorporating the list of selectable items of the overview panel. 2. Apply a “One-Window Drilldown” patter (ID 16) in order to open up a new screen 3. Apply a “List Builder” pattern (ID 13) for showing the respective details.
Illustration	Please refer to figure 3.

Another example is the German weather service wetter.de [22] where users can request weather forecast information for individual regions or cities. Both the PC and the mobile version identically request the related user input by respectively applying the “Keyword Search” pattern (ID 12). Figure 4 illustrates on its right hand side the search results for the city of Augsburg (Germany) in the desktop PC manner.

On the left side it is shown the respective output as it appears on the mobile device. For demonstration purposes the entire virtual mobile page is displayed while the black rectangle indicates the visible part on the physical screen. The user has the option to scroll up and down in order to see the whole stuff.

On the desktop PC page the content is provided in a “Tab Menu” pattern (ID 20) style for different weather forecast periods. Each panel is organized according to the “Tiled Sections” pattern (ID 22). The columns of the 3-day forecast tab contain the relevant information for today, tomorrow and the day after tomorrow. In the upper

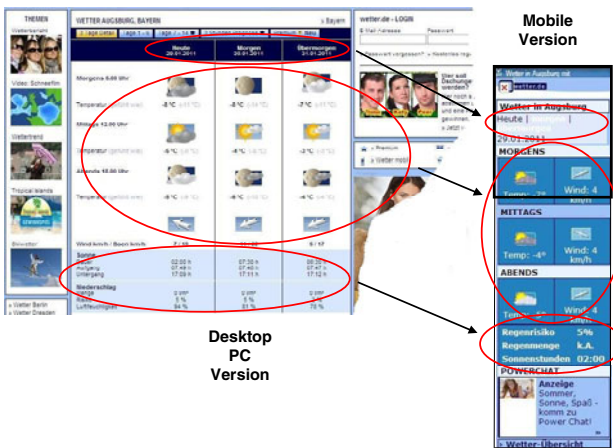


Fig. 4. Desktop PC and Mobile Version of the Weather Service wetter.de

rows of the matrix you see icons and temperature values for morning, noon and evening. The lower rows provide additional daily details about sunshine duration and precipitation.

Within the mobile version the different tabs have no equivalent because solely a forecast period of three days is available. The column headings of the “Tiled Sections” pattern are represented according to the “Permanent Horizontal Menu” pattern (ID 17). Though the horizontal menu disappears when scrolling down, it is located on each of the result pages and therefore is a permanent menu. The rows for morning, noon and evening values are translated into a type of “List View” pattern (ID 14) appearance. A selection of the daily detail measures are also displayed on the basis of the “List View” pattern.

Based on this actual situation the following three transformation patterns can be defined.

Table 4. Definition of Transformation Pattern #3

Attribute	Description
ID	T-003
Problem	How to transform a “Keyword Search” pattern (ID 12) under the given context?
Context	Transformation as contemplated in pattern T-001
Solution	Just apply a “Keyword Search” pattern (ID 12)
Illustration	-

In a similar manner it is also possible to transform specific user interfaces for other contexts of use, e.g. to create different UI for novice and expert users. Another idea is to adapt UIs for visually handicapped (not blind) people where the UI works in general similar to a UI for small screen sizes, but the elements and artifacts are displayed on a large screen in a magnified manner.

Table 5. Definition of Transformation Pattern #4

Attribute	Description
ID	T-004
Problem	How to transform a “Tab Menu” pattern (ID 20) under the given context?
Context	Transformation from desktop PC to smartphone UI as contemplated in pattern T-001
Solution	Pick the most important tabbed panel and apply a “One-Window Drilldown” pattern (ID 16) for arranging the related content onto the resulting new page. Let all other tabbed panels unconsidered.
Illustration	Please refer to figure 4

Table 6. Definition of Transformation Pattern #5

Attribute	Description
ID	T-005
Problem	How to transform a “Tiled Sections” pattern (ID 22) under the given context?
Context	Transformation from desktop PC to smartphone UI as contemplated in pattern T-001
Solution	1. Apply a “Permanent Horizontal Menu” pattern (ID 17) and use the column headings of the “Tiled Section” pattern as menu items. 2. Apply a “List View” pattern (ID 14) and incorporate the content pieces from the tiled sections. If applicable, define groups of contents and apply “List View” patterns for each of the groups.
Illustration	Please refer to figure 4

4 Conclusion

In this paper we have described a pattern-based method to transform user interfaces of interactive systems to diverse contexts of use. The main focus of our current work lies on modeling necessary relationships between the various patterns and defining a comprehensive set of transformation patterns, i.e. the rules required for pattern-driven transformation of existing abstract and semi-abstract UI models to concrete context-related target user interfaces. In a subsequent step we will put our emphasis on automation issues of the transformation process.

References

1. Alexander, C., Ishikawa, S., Silverstein, M.: A pattern language. Oxford University Press, Oxford (1977)
2. Breiner, K., et al. (eds.): Proceedings of the 1st International Workshop on Pattern-Driven Engineering of Interactive Computing Systems – PEICS 2010, Berlin, Germany, pp. 24–27 (2010)

3. Coram, T., Lee, J.: A Pattern Language for User Interface Design, <http://www.maplefish.com/todd/papers/experience.html>
4. Deng, J., Kemp, E., Todd, E.G.: Managing UI Pattern Collections. In: Proc. CHINZ 2005, Auckland, New Zealand, pp. 31–38. ACM Press, New York (2005)
5. Engel, J., Martin, C.: PaMGIS: A Framework for Pattern-based Modeling and Generation of Interactive Systems. In: Proceedings of HCI International 2009, San Diego, USA, pp. 826–835 (2009)
6. Engelberg, D., Seffah, A. (eds.): Design Patterns for the Navigation of Large Information Architectures. Proceedings of 11th Annual Usability Professional Association Conference, Orlando, Florida, July 9–12 (2002)
7. Fincher, S., et al.: Perspectives on HCI Patterns: Concepts and Tools (Introducing PLML). In: CHI 2003 Workshop Report (2003)
8. Gamma, E., Helm, R., Johnson, R., Vlissides, J. (eds.): Design Patterns: Elements of Reusable Object Oriented Software. Addison-Wesley, Reading (1995)
9. Guerrero Garcia, J., Vanderdonckt, J., Gonzalez Calleros, J.M., Winckler, M.: Towards a Library of Workflow User Interface Patterns. In: Graham, T.C.N. (ed.) DSVIS 2008. LNCS, vol. 5136, pp. 96–101. Springer, Heidelberg (2008)
10. Infragistics, Inc., Quince, <http://quince.infragistics.com>
11. Javahery, H., Seffah, A., Engelberg, D., Sinnig, D.: Migrating User Interfaces Across Platforms Using HCI Patterns. In: Multiple User Interfaces, pp. 241–259. Wiley & Sons, Chichester (2004)
12. Marcus, A.: Patterns within Patterns. Interactions 11(2), 28–34 (2004)
13. Martin, C., Roski, A.: Structurally Supported Design of HCI Pattern Languages. In: Jacko, J.A. (ed.) HCI 2007. LNCS, vol. 4550, pp. 1159–1167. Springer, Heidelberg (2007)
14. Myers, B.: Why are Human-Computer Interfaces Difficult to Design and Implement? Carnegie Mellon School of Computer Science Technical Report, no. CMU-CS-93-183 (1993)
15. Paternò, F.: Model-based Design and Evaluation of Interactive Applications. Springer, London (2000)
16. Seffah, A., Gaffar, A.: Model-based User Interface Engineering with Design Patterns. J. of Systems and Software 80(8), 1408–1422 (2007)
17. Tidwell, J.: Designing Interfaces, Patterns for Effective Interaction Design, <http://www.designinginterfaces.com>
18. Tiedtke, T., Krach, T., Martin, C.: Multi-Level Patterns for the Planes of User Experience. In: Proc. of HCI International, Las Vegas, Nevada USA, July 22–27. Theories Models and Processes in HCI, vol. 4, Lawrence Erlbaum Associates, Mahwah (2005)
19. van Welie, M.: Patterns in Interaction Design, <http://www.welie.com>
20. Homepage of the German Public TV Station ARD, <http://www.ard.de>
21. Homepage of t-online, an Online Service of Deutsche Telekom AG, <http://www.t-online.de>
22. Homepage of a German weather forecast portal, <http://www.wetter.de>