Improving the Search for User Interface Design Patterns through Typed Relationships

Jordan Janeiro[1,*], Simone D.J. Barbosa[2], Thomas Springer[1], and Alexander Schill[1]

[1] Technische Universität Dresden (TU Dresden), Department of Computer Science, Institute of System Architecture, Dresden, Germany
{jordan.janeiro,thomas.springer,alexander.schill}@tu-dresden.de
[2] Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Informatics Department, Rio de Janeiro, Brazil

Abstract. Despite being a set of proven, well-documented, contextualized recommendations for solving frequently occurring user interface design problems, user interface design patterns are still not widely used. We believe this is due to the lack of tools to help designers find patterns and identify how they can be combined to solve user interface design problems. This paper proposes to classify and make explicit the relationships between user interface design patterns. We conducted a small-scale study that indicated that this proposal is more efficient and better accepted by the participants than browsing through a user interface design library.

Keywords: user interface design patterns, semantic relationships, design pattern libraries, user interface design.

1 Introduction

Design patterns are a set of proven, well-documented, contextualized recommendations for solving frequently occurring problems. According to Alexander [1], each pattern describes a recurrent problem in a certain context and “the core of the solution to that problem (...) required to solve the stated problem, in the stated context”. Design patterns have been brought from Architecture [1] to Computer Science, first to the field of Software Engineering [2] as general software design patterns, and then to Human-Computer Interaction [3][4][5][6] as user interface design patterns.

User interface design patterns (UIDP) emerge from successful experiences with the construction and evaluation of user interface and interaction design solutions to certain real-world problems in certain domains and contexts of use. Having been recognized as successful patterns, they are usually collected and organized in pattern libraries, such as Jenifer Tidwell’s pattern collection [7] or the Yahoo! Design Pattern Library [4].

[*] Please note that the LNCS Editorial assumes that all authors have used the western naming convention, with given names preceding surnames. This determines the structure of the names in the running heads and the author index.

Despite their reputation, UIDPs are still not widespread among software developers. One of the key reasons for that is the lack of tools to support their usage, like for example finding a pattern which solves a certain problem, or even identifying a set of patterns which can be used together to solve a more complex user interface problem.

We believe that a first step towards such a tool would be to provide explicit typed relationships between patterns, allowing designers to quickly identify which of them may or may not be used together, and how to combine them to address more complex problems. In typical UIDP libraries, the types of relationships between patterns are usually left implicit in the UIDP description.

An analysis of the library of van Welie [6], for example, illustrated by Figure 1, shows a high number of relationships between the patterns; each vertex represents a design pattern and the edges represent the relationships between two UIDPs. Such number of relationships demonstrates the possibilities to form many combinations of compositions with UIDPs. However, the creation of the compositions is still not possible, because the current libraries only express the relationships as links between the patterns, and no further information about the meaning of the relationship is described. Thus, whenever a user needs to identify the type of relationship between UIDPs, she needs to read the complete textual descriptions and interpret the type of the relationships that are only implicitly expressed in the description.

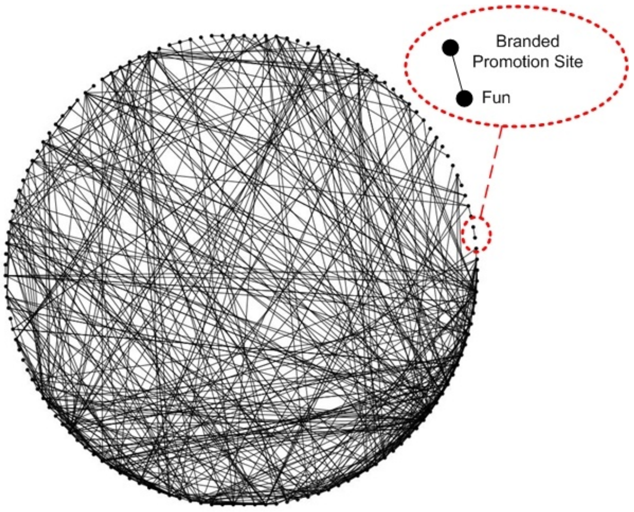


Fig. 1. Graphical Representation of the Relationships between User Interface Design Patterns in a Typical Library

In this paper, we propose some classifications to help identify the relationships between UIDPs (Section 4), making it possible, in future work, to create a tool that uses them to support the user in identifying not only individual patterns, but how they may be combined to handle complex user interface problems.

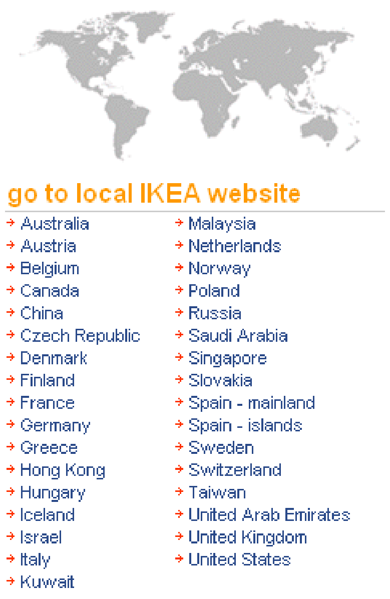
We have validated the kinds of relationship types we propose (Section 5) asking a group of users to classify pattern relationships found in a UIDP library in the proposed relationship types. Finally, we evaluated the usefulness of such classification (Section 6), requesting a group of users to report their experiences using the typed relationships in searching for UIDPs which should solve a given problem.

2 User Interface Design Patterns

User interface design patterns (UIDPs) are well-documented user interface and user-system interaction solutions for known and frequently occurring user interface problems. These solutions have been shown to benefit end users; they have already been implemented, evaluated, and proven successful.

Currently in the literature there are sets of examples of UIDPs, representing the user's perspective on using a system and the designer's perspective on building a system. Independently of the perspective, both kinds of UI design patterns benefit somehow the system's end user, providing well-established, familiar solutions to widely known problems. Among the major UIDP libraries, we find: Tidwell's patterns for effective interaction design [7], van Welie's pattern library for interaction design [6], and the Yahoo! Design Patterns Library [4]. Some other libraries target a more specific application domain, such as: user interface design patterns for games [8] and for mobile user interfaces [3].

For illustration purposes, let us consider the country selector pattern, described in [6]. Such a pattern belongs to the making choices category and it targets the problem of allowing the user of a website to navigate from the company's global website to a country-specific website, as illustrated by Figure 2.


go to local IKEA website

- Australia
- Austria
- Belgium
- Canada
- China
- Czech Republic
- Denmark
- Finland
- France
- Germany
- Greece
- Hong Kong
- Hungary
- Iceland
- Israel
- Italy
- Kuwait
- Malaysia
- Netherlands
- Norway
- Poland
- Russia
- Saudi Arabia
- Singapore
- Slovakia
- Spain - mainland
- Spain - islands
- Sweden
- Switzerland
- Taiwan
- United Arab Emirates
- United Kingdom
- United States

Fig. 2. Example of the Country Selector User Interface Design Pattern

As sometimes international websites function as a proxy to country-specific websites, it is necessary that the website allow the user to navigate easily to her specific location. It is also important to state that this pattern is not the language selector pattern, which allows users to define the preferred language for the website. To provide a better solution in this case, both design patterns should be combined and used, because users interested in a certain location, like tourists, may not speak its local language.

Defining a user interface design pattern library is already a significant advance on the standardization and definition of best practices for building user interfaces. However, each library uses its own descriptive structure for its patterns, distinct if ever so slightly from one another. There is no standardization here, i.e., currently there is no consensus about a semi-structured standard description for user interface design patterns. For example, in [6] a pattern is described through the problem it targets, the description of the solution, the situation in which it may be used, how it is used and why a designer should use it to solve the targeted problem. In [4] a pattern is described by its title, the problem it solves, its context of use and the solution described by the pattern to solve its problem, without detailing how or why.

Besides the proposal of different structures by the libraries to describe patterns, another problem which arises is the lack of clear meaning of the relationships between patterns. Each pattern is often used together with other patterns; however, none of the design pattern libraries clearly describes how the patterns relate to each other, i.e., the explicit relationships (with clear semantics) between them.

3 User Interface Design Pattern Relationships

The main idea of the UIDPs is to apply the concept of separation of concerns, explored by the software engineering design patterns, creating components of user interface which aim to solve specific user interaction problems. Therefore, with such a concept it may be possible to compose a certain user interface solution through such components, instead of creating it from scratch.

Compositions between UIDPs are already informally documented by the user interface design pattern libraries, allowing identifying which UIDPs are commonly used together in certain situations. For example, it is specified in the van Welie library that the design pattern Accordion is often used within the Main Navigator design pattern.

The Pattern Language Markup Language (PLML) [9] is a proposal to standardize the documentation of user interface design patterns. This language describes important information as: the problems a UIDP solves, the solution the patterns propose and the relationship between the design patterns. This last property of the language is particularly important in the context of this paper, because it represents an explicit description of the relationship between patterns, which in our case is a basic concept to support the composition of UIDPs. An example of these kinds of relationships is described in the next excerpt of code of PLML. In such code, the pattern *Accordion* from van Welie's library, relates to the pattern *Closable Panels* from Tidwell's library.

Example of Relationship Description between the User interface Design Patterns Accordion and Closable Panels

```
<pattern-link type="is-a" patternID="Closable_Panels"
collection="tidwell" label="Closable Panels"/>
```

As illustrated by this example, through PLML it is possible to specify four properties in describing a relationship between patterns: *patternID*, which references the ID of the targeting UIDP from the relationship (Closable_Panels); *collection,* which describes to which library the pattern belongs (tidwell); *label,* which describes the label of the relationship between the patterns (Closable Panels) and *type* (is-a), which describes the semantic of the relationship between the two patterns. However, such feature is still not explored to be used in the current description of UIDPs.

Despite the capability of PLML to describe the types of relationships between UIDPs, the language only supports three types: *is-a*, *is-contained-by* and *contains* [9]. The first type of relationship, despite implying a generalization-specialization hierarchy, actually means that two design patterns are equivalent to each other, meaning that a designer can exchange the use of a UIDP with another in a certain context. The other two types are related to the composition of UIDPs; the *is-contained-by* type refers to a design pattern which is used within another UIDP, whereas the *contains* type refers to the design pattern which uses another UIDP.

On the language specification such a small set of types are may not be enough to describe more precisely the semantics of the relationships between the design patterns. Therefore, we intend to present an extension to these existing types to allow PLML to describe more precise the semantics of the relationships between the design patterns.

4 User Interface Design Patterns' Semantic Relationship

As object of our analysis we used the UIDP Library from Martjin van Welie [6]. Our study consists of identifying the relationships between the UIDPs and to identify other sorts of relationship between design patterns, based on the context they are proposed to be used and their relationship descriptions of PLML.

Based on such analysis we identified seven types of semantics that we propose to extend the current defined types by PLML. They are the following:

- **Used With:** describes that a UIDP x can be used together with another UIDP y and they are not hierarchically related, such as in a composition relationship. Such type of a relationship is illustrated in the library of van Welie by the following text excerpt: "*… Paging is also often used together with a List Builder …*". By this description we can identify that the Paging and List Builder patterns are independent but they are combined in this case through the *used with* relationship.
- **Equivalence:** describes that a UIDP x is like a UIDP y, making it possible to use both to solve the same problem in a certain context. Such type of a relationship is illustrated in the library of van Welie by the following text excerpt: "*… Accordions are alternative to Navigation Tree …*". By this description we can identify that in a certain situation both of the patterns can be used as alternatives to solve the same problem.

- **Similarity:** describes that a UIDP x has some characteristics similar to the UIDP y but, depending on the specific problem they target, one of the patterns should be slightly adapted. "… *the site has got some type of Main Navigation that allows* …". In this description, the similarity relationship is between the Breadcrumbs and the Main Navigation design patterns, which can be used as an alternative to each other in a certain specific context, if some slight modifications are performed.
- **Conditional Equivalence:** describes that a UIDP x may be considered to be equivalent to a UIDP y, if a set of conditions are satisfied. Such type of a relationship is illustrated in the library of van Welie by the following text excerpt: "… *If used for navigation it is (Accordion) conceptually equivalent to Tabs* …". In this description we can identify that the Accordion pattern can also be used as an alternative to the Tabs pattern only if it is used for navigation purpose.
- **Realization:** describes that a UIDP x implements the concepts described by a UIDP y. Such type of a relationship is illustrated in the library of van Welie by the following text excerpt: "… *Accordions can be a good way to implement a Frequently Asked Questions (FAQ)* …". In this case, the FAQ is a recommendation of a functionality which is desirable to have in an application, not an implementation of a concrete user interface component, as in the case of the Accordion pattern. Therefore, we established this kind of relationship between both patterns because the Accordion can implement the functionalities described by the FAQ.
- **Enhancement:** describes that a UIDP x builds upon an UIDP y, enhancing its functionalities. Such type of a relationship is illustrated in the library of van Welie by the following text excerpt: "… *This pattern (Advanced Search) builds on the Search Box pattern by adding some more search options* …". By this description we assumed that the Advanced Search pattern enhances the Search Box pattern with more options, which improves the search function of an application.
- **Conflict:** describes that a UIDP x and a UIDP y must not be used together. Such type of a relationship is illustrated in the library of van Welie by the following text excerpt: "… *Although Accordions are often used as part of a Wizard I strongly recommend against it since it is worse than regular implementations from a usability point of view* …". By such description we understand that although it is really common to use the Accordions pattern as part of Wizard pattern, the author discourage such a practice because the Accordion does not handle properly the navigation issues in Wizard-style applications.

5 Semantic Relationship Assignments

In this section, we describe how we have defined the set of proposed relationship types and the techniques we used to validate them.

Our methodology to extract and evaluate the proposed design pattern relationships consists of two steps: (i) the definition of the semantics of the relationship types, presented in the last section and (ii) the assignment of such semantics to the relationships of the design patterns.

In the first step we analyzed the UIDP descriptions and their references to other patterns, which helped us to select a set of representative labels for classifying the relationships with certain semantics. This list was already presented in the last section as an extension to the element *pattern-link* of the PLML language, to enhance the description of relationships between UIDPs.

In the second step our goal was to validate the proposed relationship semantics we defined. Our approach consisted of proposing to the users to assign the semantics we defined to existing UIDPs´ relationships, already described by the PLML language.

We selected 15 design patterns from the van Welie library to be evaluated. As the library is divided in 15 categories of patterns, we decided to select a UIDP of each of them. All of these patterns together contain 71 relationships, which should be classified by the users. The users should read the text description of a pattern, containing links to other patterns, and classify these links using the proposed relationship types.

We performed the experiment with five users, two men and three women all of whom are computer science professionals, and whose ages vary between 22 and 24 years old.

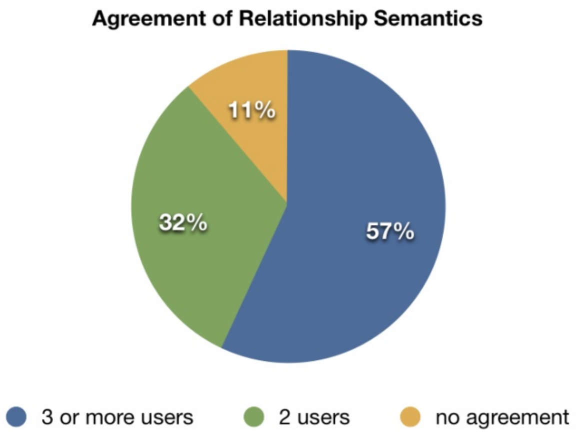


Fig. 3. Agreement of Users in Classifying a Relationship between Two User Interface Design Patterns

Figure 3 presents the agreement, in percentage, of the users in assigning semantics to relationships between design patterns. According to the chart, in 57% of the assignments, three or more users assigned the same type to a relationship; in 32% of the assignments, two users assigned the same type to describe a relationship; and in 11% of the relationships, none of the users' assignments matched.

We performed such experiment with a group of users to obtain the agreement of the majority in assigning certain semantics to the relationships of the design patterns. Therefore, if the majority of the users agree with the semantics to a relationship, it means that there is a higher probability that the chosen semantics represents a suitable term to describe the relationship.

As the assignments are important information to us, we analyzed carefully the three categories we registered in the experiment: the agreement of three or more users, the agreement of two users and no agreement. For the first case, we automatically added the assignments to our database, because the majority of the users who performed the experiment agreed in a common relationship type, which for us means that the assigned meaning sufficiently characterizes that relationship. In the second case, we included our own analysis as an additional assignment, i.e., we only considered the assignments of the two users if we agreed with them. For the relationships in the third case, we analyzed by ourselves the relationships to classify them.

6 Evaluation

In order to evaluate the effectiveness of providing typed relationships in searching and finding user interface design patterns, we prepared two usage scenarios.

In the first scenario, the participant should build a form for booking flights, and in the second scenario, the participant should develop a form for registering personal information in an e-commerce web site. For each of these scenarios, participants should select the user interface design patterns which they thought were the most suitable to build their user interfaces. For each scenario, participants had to search for the UIDPs using a different resource. We selected two different resources for the experiment: van Welie's user interface design pattern library "as is" (*Plain Library*); and a tool we implemented to help browse the library using the typed relationships between patterns (*Navi Browser*).

We selected a group of 8 participants, 6 men and 2 women, all of them professional computer scientists.

To reduce the bias from the order in which the tasks were performed and the resources used, we have distributed the participants in four groups, varying the order of tasks and resources, as presented in Table 1.

Table 1. Task and Resource Groupings Used to Reduce Bias in the Experiment

First Performed Task	Second Performed Task
Flight Booking / Plain Library	Personal Registration / Navi Browser
Flight Booking / Navi Browser	Personal Registration / Plain Library
Personal Registration / Plain Library	Flight Booking / Navi Browser
Personal Registration / Navi Browser	Flight Booking / Plain Library

After performing the proposed tasks for our evaluation, the participant should answer a few questions about their experience with each resource. Each question could be answered using a 5-point scale, ranging from 1 (lowest/worst) to 5 (highest/best). The questions proposed in the form are following:

- Q1: How easy was it to find the UIDPs using the proposed application?
- Q2: How adequate were the UIDPs to the user's solution?
- Q3: How easy was it to identify the relationships between the UIDPs?

- Q4: How accurate were the relationships between the UIDPs?
- Q5: How useful were the relationships between the UIDPs?
- Q6: How much time do you think the task took to be accomplished?
- Q7: How much did the application help to reduce the time to obtain the UIDPs to design the solution?
- Q8: What is your overall impression of using the proposed method to find the UIDPs?

From the experiments, we could identify that all of the users achieved the patterns for their tasks following a certain procedure: they searched first, in the list of all UIDPs, for patterns whose title contained keywords for the problems they needed to solve, selecting these for later detailed analysis; then, they analyzed all of the patterns linked to the ones within their list. Therefore, in the context of this experiment, the participants identified first the patterns *Forms* and *Booking* for the flight booking scenario, and *Forms* and *Registration* for the personal registration scenario. From there, participants identified additional patterns through the typed relationships, as presented in Table 2.

Table 2. User Interface Design Patterns Selected in Each Scenario

Flight booking		**Personal registration**	
Forms	Booking	Forms	Registration
Advanced Search	Advanced Search	Advanced Search	Form
Constraint Input	Homepage	Booking	Login
Grid-Based-Layout	Processing Page	Constraint Input	Product
Input Error	Purchase Process	Grid-Based-Layout	Recommendation
Message	Search Results	Input Error	Shopping Cart
Login	Wizard	Message	Wizard
Wizard		Login	

We analyzed the collected data to understand the experience of the users in searching for user interface design patterns connected through the typed relationships. Figure 4 presents in a chart the comparison of the user satisfaction in using both applications for this experiment, meaning indirectly a comparison between the user satisfaction in making use of the semantics of the relationships and not.

The chart shows that using typed relationships was consistently rated better than the plain library. Questions 3 and 5 present valuable information to our approach, because they specifically evaluated the participants' satisfaction in using the typed relationships for searching design patterns. In the answers for these questions, the score for our approach is quite higher than the plain library, suggesting that the use of typed relationships makes the search process easier. During the interviews, the participants mentioned that they assigned a high score to our approach because they did not have to read the entire descriptions of the design patterns to understand the way they relate to each other, thus, they could save time in interpreting the meaning of the relationships and deciding whether other related design patterns should also be considered in the solution.

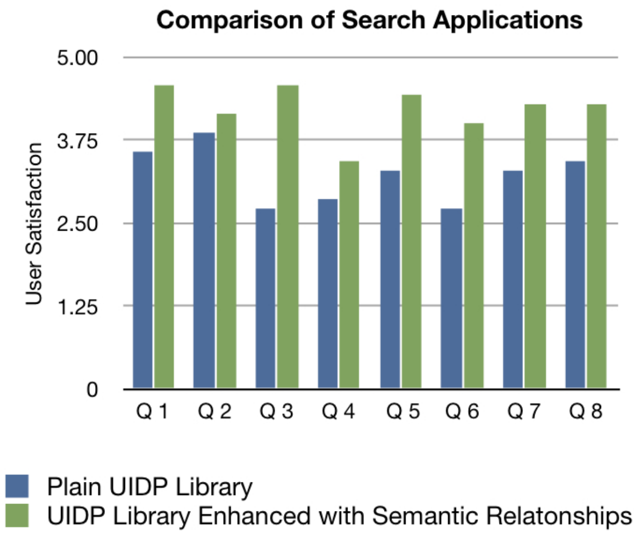


Fig. 4. Evaluation of Using Typed Relationships to Search for User Interface Design Pattern

7 Related Work

The Pattern Language Markup Language (PLML) [9] was developed to describe user interface design patterns in a standardized way, through a set of attributes. Van Welie has used PLML in his library. As we have seen, the types of relationships between patterns are very limited in PLML. Moreover, there is no established research work that proposes to use the information described by the semantic relationships to support the development of a search mechanism to retrieve design patterns more efficiently. In our approach, we have defined these new types of relationships between patterns as a first step towards such a search mechanism.

Previous efforts have investigated the use of user interfaces guidelines to support the process of developing applications [10]. However, the core problem of using guidelines is still the lack of tools and methods for describing and validating them, specially regarding to high-level guidelines. Our approach combines the contextualized information provided by the user interface design patterns with clear semantics of their relationships that can be used to develop a tool to support using patterns in user interface design.

Our underlying vision is somewhat akin to Semantic Web ideas introduced in [11]. In such work, the authors discuss the ideas of enhancing the Web through the attachment of additional information that describes the semantics of a resource in the Web. In the realm of web services, the World Wide Web Consortium (W3C) has proposed a standard for the Semantic Annotations for WSDL (SAWSDL) [12]. Such standard defines a set of attributes to annotated Web Services to better describe their semantics. Although such approach allows the attachment of any kind of semantic descriptions, the most common use and support focuses on the functionality descriptions of web services, and there is no support specifically defined to describe the user interfaces aspects associated to it. In our approach, we share the ideas introduced in the semantic description of web services to perform more refined searches, but our main goal is to model and support information related specifically to user interface issues, both problems and solutions.

Regarding the execution of semantic searches, there are many existing semantic service matchmakers which perform discovery processes based on the attached semantic, by non-logic, logic or hybrid matching techniques [13]. In future work, we intend to extend such matchmakers to perform matches of user interface design patterns to a specified user interface problem described by the user performing a search.

8 Conclusion

This paper proposes a new set of semantic types which enhance the description of user interface design pattern relationships. The proposed types were used by a group of participants to describe different kinds of relationships between UIDPs. We then conducted an experiment with another group of participants, in which we have identified the effectiveness of using the typed relationships to help search for design patterns. In our experiment, we compared two kinds of resources for finding user interface design patterns: without explicitly typed relationships, as it is currently available in the design pattern libraries, and with the explicitly typed relationships. The experiment showed that the typed relationships provide a better support to the search, because they make it easier and more efficient to determine the relationship between UIDPs, and thus to identify which sets of design patterns are suitable to solve a certain user interface problem, and how they can be combined.

We believe that our proposal of typed relationships can further be used to implement a new kind of UIDP search engine. Instead of browsing a design pattern library, interpreting and selecting compositions of UIDPs, the user should be able to perform such tasks semi-automatically; the search engine we envision will let the user specify a keyword-based query to describe her user interface problem, and the engine will be able to perform the search for such compositions. The goal is that the search engine customizes the recommendation of a set of design patterns for each situation, leveraging thus a new supported method for solving user interface problems.

Acknowledgments. We thank our colleagues at TU Dresden and at PUC-Rio for invaluable discussions related to this work. Simone Barbosa thanks the financial support of CNPq (557.128/2009-9) and FAPERJ (E-26/170028/2008). Jordan Janeiro, Thomas Springer and Alexander Schill thank the supporting of project CRUISe and the financial support of the Federal Ministry of Education and Research of Germany (BMBF).

References

[1] Alexander, C., Ishikawa, S., Silverstein, M.: A Pattern Language: Towns, Buildings, Construction. Oxford University Press, Oxford (1977)

[2] Gamma, E., Helm, R., Johnson, R.E.: Design Patterns. In: Elements of Reusable Object-Oriented Software. Addison-Wesley Longman, Amsterdam (1995)

[3] Mobile User Interface Design Patterns, `http://patterns.littlespringsdesign.com/index.php/Main_Page`

[4] Yahoo! Design Pattern Library, `http://developer.yahoo.com/ypatterns/`

[5] Borchers, J.O.: A Pattern Approach to Interaction Design. In: Proceedings of the 3rd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques, pp. 369–378. ACM, New York (2000)

[6] van Welie, M.: Welie.com - Patterns in Interaction Design, `http://www.welie.com`

[7] Tidwell, J.: Designing Interfaces: Patterns for Effective Interaction Design. O'Reilly Media, Inc., Sebastopol (2005)

[8] Bjork, S., Holopainen, J.: Patterns in Game Design, Charles River Media (2004)

[9] Fincher, S.: Perspectives on HCI Patterns: Concepts and Tools, Introducing PLML (2003)

[10] Cohen, A., Crow, D., Dilli, I., Gorny, P., Hoffman, H., Iannella, R., Ogawa, K., Reiterer, H., Ueno, K., Vanderdonckt, J.: Tools for Working with Guidelines. SIGCHI Bull. 27, 30–32 (1995)

[11] Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American Magazine

[12] Kopecky, J., Vitvar, T., Bournez, C., Farrell, J.: SAWSDL: Semantic Annotations for WSDL and XML Schema. IEEE Internet Computing 11, 60–67 (2007)

[13] Klusch, M.: Semantic Web Service Coordination. In: CASCOM: Intelligent Service Coordination in the Semantic Web, pp. 59–104. Birkhäuser, Basel (2008)