SCENARIO-BASED DESIGN IN DESIGN PATTERN MINING

Claudia Iacob
University of Milan, Italy

ABSTRACT

Design patterns are tools to support social creativity in that they allow communities of designers to make available knowledge related to design experiences, such as problems, solutions and design decisions. Identifying design patterns is a process known as design pattern mining. This paper describes one of the techniques used throughout a series of design workshops held for identifying design patterns in the design of software applications for synchronous collaboration. The paper describes the way the technique was applied in concrete design cases, and its influence on the results obtained throughout the workshops. Results indicate that the use of scenario-base design supports the exploration of the design space of the application under design. This allows the generation of a larger number of scenario ideas to support the design pattern mining process.

Keywords: Design patterns, design workshops, scenarios

1. INTRODUCTION

Design patterns provide a way of capturing and sharing knowledge related to design problems. Each design pattern is a multimedia document storing “a proven solution to a recurring design problem” [1]. Design pattern mining processes have as goal the identification of design patterns in the design of software applications. A wider exploration of the design space considered leads to a broader spectrum for the patterns’ identification. Scenarios – defined as stories about actors (in this case, the potential users of the applications) and the activities (the actions the application supports) they perform when engaged in the interaction with the application under design [2] – have a great potential in supporting the design space exploration. Moreover, scenarios support designers’ communication and common understanding.

This work argues that scenario-based design [3] - i.e. the technique of using scenarios during design processes - supports design pattern mining processes in that it allows designers to create design situations and to find solutions to tackle them. To support this argument, the technique has been applied during a series of design workshops through which teams of software designers were asked to design software applications for synchronous collaboration. The goal of the investigation was to analyze the impact of using scenario-based design in design pattern mining processes. Eight teams designed software applications to support collaborative drawing, collaborative text editing, collaborative puzzle, and crosswords solving. During the first phase of the process they followed, participants were asked to generate as many scenario ideas as they could consider for the design of the application they chose. These scenario ideas were used for identifying commonly recurring design problems, solutions, and decisions in such design processes. This lead to the identification of a set of good practices which were further on documented through design patterns.

The paper is structured as follows: section 2 provides an overview of the scenario-based design technique and of the design patterns as tools to support knowledge management; section 3 describes the workshops conducted in terms of the participants, the procedure they followed, the results obtained and the set of design patterns identified; lastly, section 4 draws the conclusions.

2. BACKGROUND

2.1 Scenario-Based Design in Software Design

“Software design is fundamentally about envisioning and facilitating new ways of doing things and new things to do.” [3] The complexity of software design problems exceeds one’s individual ability to tackle them and asks for the collaboration of stakeholders with different expertise and backgrounds

[4]. Graphic designers, software engineers, programmers, human-computer interaction specialists, and marketing people come together and collaborate in the design of software applications [5]. In addition to that, it is often the case that software addresses the problems of clients (often, the users) which may not be (and are not willing to be) experts in software design. However, they need to communicate with the software designers which are not experts in the domain of the clients/users. Therefore, one of the challenges in designing software applications is finding ways to communicate design ideas and interaction representations.

One way to do that is by describing as in a story the actors (the potential users of the application) and the activities (the actions the application supports) they would perform when engaged in the interaction with the application under design. Such descriptions are called scenarios. They are stories which describe people in action, their goals, and motivation, the concrete descriptions of activities that engage the user when performing a specific task [2]. Scenarios prove to be powerful design tools in that:

a). they are easily understandable by all those involved in the design process – software designers, programmers, clients/users,

b). they allow reasoning about situations of use even before those situations actually exist [3],

c). they support software designers in understanding the requirements expressed by the client/user, providing a "a description sufficiently detailed so that design implications can be inferred and transformed into actual models" [2],

d). they constitute a bridge between the specialized language of the client/user and the specialized language of the designer,

e). they provide insight into the ways to tackle usability aspects, since they provide "snapshots" of the application in use.

Defining a scenario should answer a set of pre-requisites. First, any scenarios should have a narrative character; it should sound like a story. This not only supports the communication and common understanding among the stakeholders involved in the design process, but it also supports the dialog of the designers with the clients/users and vice versa. In addition to that, any scenario should answer a set of questions, such as: who are the users?, what are their goals?, what is their motivation to use the software application?, how could they use the application, and when and where can the application be used?.

Who are the users?

Any scenario provides snapshots of users in action. Hence, a scenario should provide a profile of the user, able to help the designers formulate a set of usability requirements for the application and target it to the right audience.

What are the users' goals and motivation?

Any user has goals and objectives [3]. This is what the user wishes as an outcome of his/her interaction with the application. Information on the goals and motivation of the users may translate into additional requirements for the application's design.

How could the users use the application?

Just as stories, scenarios have a plot. The users achieve their goals by following a set of steps, a procedure. Information on the interaction process followed by the actors provides insight into the functional requirements of the application.

When and where could the application be used?

Time and place are always revealed in stories. Most of the time, they are the most significant piece of information, setting the context of the overall story. Scenarios follow the same pattern in that they provide information on issues related to time (when the application can be used) and space (where the application can be used). This information supports the decision of the platform on which the application will be materialized (eg. mobile devices, laptops).

Scenario-based design – i.e. the technique of using scenarios during design processes – has been applied in various stages of the software development cycle, such as requirements analysis, user-designer communication, design rationale, envisionment, documentation and training, evaluation, abstraction and team building [2]. This work describes the application of scenario-based design in processes of mining for design patterns in the design of software for synchronous collaboration. The aim of the investigation is to check if scenario-based design supports designers in exploring the design space of the application and what is the impact of this exploration on the design pattern mining process.

2.2 Design Patterns

Design patterns provide a way of capturing and sharing knowledge related to design problems. Each design pattern is a multimedia document storing “a proven solution to a recurring design problem” [1]. Different authors proposed different templates for defining design patterns. These templates generally include the name of the design pattern, the description of the problem it addresses together with the forces (i.e. the consequences and secondary implications of the problem) that influence this problem, some examples of situations in which this problem can be met and a possible solution to tackle the problem [6].

The notion of design pattern was first proposed by Alexander [7] who described patterns as tools for capturing and making available and communicable knowledge related to architectural design. His approach had a wide impact in several domains, including software engineering and Human-Computer Interaction (HCI). On one hand, software engineering applied design patterns for expressing Object-Oriented software design experience. On the other hand, HCI designers adopted the design pattern approach to document and describe “the reasons for design decisions and the experience from past projects, to create a corporate memory of design knowledge” [1].

Several collections of design patterns [8, 9] for interface and interaction design are now available in the Web. A collection of patterns targeted for the design of social interfaces is introduced in [10]. The focus of these patterns is on the design of systems which support social activities like: broadcasting and publishing, collecting data, rating, or collaborative editing. Moreover, several collections of patterns have been proposed for the design of groupware technology [11], and cross-culture collaboration [12]. Even if synchronous collaboration is common in various contexts (such as searching [13], or sketching [14]), little work has been done in identifying design patterns for the design of systems for such collaboration.

Design patterns proved useful in various fields in that they support knowledge management processes such as knowledge creation, knowledge integration, and knowledge dissemination.

Design Patterns and Knowledge Creation

Knowledge creation requires the externalization of one individual’s tacit knowledge or on-the-go experience which is “not a matter of emptying out the mind but of actively reconstructing it, forming new associations, and expressing concepts in external representations while lessening the cognitive load required for remembering them” [5]. This implies moving from vague mental conceptualizations of an idea to a more concrete representation of it. Externalization provides means for others to interact with, react to, negotiate around, and build upon one’s idea. It is an opportunity for creating a common language of understanding [4]. Design patterns provide a semi-formal template for defining a design problem, together with its solution, applications and utility. The externalization of ideas through an existing template of definition not only facilitates each designer’s externalization process, but also supports knowledge creation. Design patterns become in this way an artifact, used to express knowledge related to a design problem and to “communicate and facilitate shared understandings across spatial, temporal, conceptual, or technological gaps” [15].

Design Patterns and Knowledge Integration

Knowledge integration implies “identifying and evaluating the interaction between new and existing knowledge” [16]. One of the challenges to be overcome during the process of knowledge integration is assessing how “new information interacts with existing knowledge because knowledge base modifications intended to correct shortcoming may conflict with existing knowledge and introduce problems”. Identifying these conflicts and resolving them are both necessary tasks to be performed during knowledge integration. Within a community of designers working together, the task of knowledge integration is given to the designer contributing to the already existing knowledge base. This is to say that it is each designer’s duty to make sure that the knowledge s/he acquired and externalized is placed in the common knowledge base and related to the already existing knowledge.

According to [17], knowledge integration comprises two tasks: i). conceptual generalization, and ii). representational formalization. The first phase asks for relating information from one context to another, while the second phase implies putting information in a form that allows its access and interpretation. Design patterns support both of these phases in that: i). they allow the creation of relationships among different design spaces, and ii). they provide a semiformal definition template in which information related to design problems can be expressed.

Design Patterns and Knowledge Dissemination

Many designers today "are seeking to reuse knowledge in new applications and to share encoded knowledge" [18]. Design decisions and experiences may constitute a corporate memory within a community of designers, to which they could relate. Sharing such knowledge supports, on the one hand, its reuse and, on the other hand, it facilitates the creation of a common language and identity within the community. Moreover, making all voices heard and allowing all design problems to be shared enables social creativity which, as defined by Fischer, "explores computer technologies to help people work together" [4]. Supporting the dissemination of knowledge is one of the major goals of a design pattern approach to design. Design patterns are artifacts used within a community as a bridge language to support the collaboration and the interaction of all the stakeholders involved in the design process.

"The scarcest resource for most of us as we try to understand and solve problems is not information; it is attention" [17]. This is to say that in some situations it is not the leak of information that causes problems in design, but the overwhelming quantity of information. Crucial in these situations becomes finding the right information at the right time and in the right format. Design patterns are described in a semiformal way by structured documents, providing designers with reading templates facilitating their readability and understandability.

3. SCENARIO-BASED DESIGN IN PATTERN MINING PROCESSES

The overall aim of this work is to identify a set of design patterns for the design of software systems which support synchronous collaboration. One step towards that was to conduct a series of design workshops during which participants were asked to design software systems to support people in performing activities such as drawing, text editing, puzzle, and crosswords solving collaboratively and in real-time. One of the techniques used during the workshops was scenario-based design. This section describes the workshops in terms of their participants, the procedure they followed and its impact on the results, the results obtained, and some of the design patterns identified.

3.1 Participants

The total number of participants was 31, out of which 84% were male, and 16% female. 17 of the participants (55%) were Master students in a "Human Computer Interaction" class (HCI). They worked in 4 teams, each team working on one of the following problems: drawing, puzzle solving, text editing, and crosswords solving. 10 participants (32%) were undergraduate students in a course on "Technologies for Collaboration" (TC). They were divided into 3 teams, and they worked on the following problems: drawing, puzzle solving, and crosswords solving. Lastly, 4 of the participants (13%) were professional designers. They worked as a team in designing an application for collaborative drawing.

Problem	Participants		Male	Female	Design	HCI	TC
	Nr.	Teams					
Drawing	11	3	8	3	4	4	3
Text editing	4	1	4	0	0	4	0
Puzzle solving	7	2	7	0	0	4	3
Crossword solving	9	2	7	2	0	5	4

Table 1 – Workshop participants' statistics

Out of the 31 participants, 35% worked on the design of applications for collaborative drawing, 13% on collaborative text editing, 23% on collaborative puzzle solving, and 29% on collaborative crosswords solving (Table 1).

3.2 Procedure

Participants worked in teams of 3-5 people and the duration of a workshop was 2 hours. Each team was presented with the list of problems and was encouraged to choose one problem for which to design a software application. Participants were initially presented with the techniques they would be using: scenario-based design, sketching, and mock-ups.

During the first phase, participants were asked to define as many scenarios as they could consider for the application under design. In defining a scenario, they would consider answering the questions: a). who are the users?, b). how can they use the application?, c). how could they achieve their goals using the application?, d). what is the motivation of their collaboration?, and e). when and where could the application be used?.

The second phase asked participants to choose another problem from the list and to find as many similarities and differences between the two problems (the one chosen during the first phase and the one chosen during the second phase). The purpose of this exercise was to identify commonalities and major differences between collaborative systems addressing different domains. Similarities would indicate the possibility of abstracting design details related to the two domains, while differences would suggest that similar design problems would require different design solutions for the two domains compared.

Lastly, participants were asked to design the GUI and the interaction process of the application related to the problem they initially chose during the first phase. For that, they were strongly encouraged to sketch their ideas, express all the design problems they encounter and, possibly, create a mock up of their overall design. They were observed throughout the process.

3.3 Problems and Scenarios

Drawing

The problem of collaborative drawing asked for the design of a software application which would allow painters, graphic designers and/or visual artists to collaboratively create one diagrammatic representation in real-time. Three teams worked on this problem. The first team generated 26 scenario ideas, including: a). networks of friends come together and draw collaboratively as in playing a game, b). drawing collaboratively and projecting the drawing in different parts of the world, c). creating a city event which brings citizens together and providing them with a recording wall for drawing, d). create an online gallery and see it as a recruiting place.

The second team generated 4 scenario ideas. Some of the ideas they generated were common to those coming from the first team. One example of common idea is allowing the application to revolve around a city event where people come together and, using different drawing techniques, draw collaboratively. Another idea proposed was seeing the overall drawing as the composition of individual drawings that each user could create in a private area of the application.

Lastly, the third team proposed 15 scenario ideas. The recurring solution for the synchronous collaboration was allowing each collaborator to draw separately and compose the individual drawings into a collaboratively created drawing.

Text Editing

The problem of collaborative text editing required participants to design an application which would allow a group of users to create a summary of a written text in a synchronous collaborative fashion. One team worked on the problem of collaborative text editing and 19 scenario ideas were generated. They included: a). the creation of a mash-up between an instant messaging system and a document editor, b). having a group of students take notes collaboratively during a lecture on a tablet PC, c). integrating social features such as ranking, tagging, annotating, commenting, d). allowing the identification of each individual’s contribution to the document, e). supporting the creation of reports on the history of the collaboration.

Puzzle Solving

The puzzle solving problem asked for the design of an application which would allow groups of users to collaboratively solve one puzzle in real time. Two teams worked on the problem of collaborative puzzle solving. The first team generated 20 scenario ideas. They envisioned several motivations users might have in solving puzzles collaboratively: a). the puzzle can be seen as a game or as an artistic act which brings people together, b). the puzzle can be used with medical goals such as helping elderly people in remembering things, c). the puzzle could be an educational tool through which pupils learn together.

The second team proposed 12 scenario ideas. They suggested including different levels of difficulty in playing the game, so that different users may choose different levels. Also, they considered including a private area in the application where users may try out pieces of the puzzle before adding them to the shared board. Scenarios also included the idea of having different users playing together, but using different devices.

Crosswords Solving

The problem of crosswords solving asked for a software application which would support a group of users in solving collaboratively and synchronously one crosswords game. The crosswords solving game was solved by two teams. The first team generated 26 scenario ideas. They described users: a). interacting with the application through vocal interactions, b). creating private groups to which their friends could join, c). organizing competitions among teams playing together, d). creating personalized crosswords, e). answering crosswords questions under a time constraint.

The second team proposed 21 scenario ideas. Some of the ideas included solving crosswords for collaboratively learning a new language, making sure that all the collaborators speak the same language, and allowing users to ask for hints for the answers.

3.4 Overall results and discussion

Design pattern mining processes have as goal the identification of design patterns in the design of software applications. A wider exploration of the design space considered leads to a broader spectrum for the patterns' identification. This work argues that scenario-based design supports design pattern mining processes in that it allows a wider exploration of the design space considered for the design.

Scenarios support designers in imagining situations of use even before they actually exist. For that, they freely come up with ideas for the design of the application considered to create possible interaction contexts. Designing these interactions, designers come across new and innovative solutions to recurring design problems. Hence, they identify best practices in design processes, which may be documented by design patterns.

During the first phase of the workshops conducted, participants were encouraged to externalize scenario ideas for the design of an application to solve one of the problems described above. These ideas answer the questions: who are the users?, what are their goals?, how could the users interact with the application?, why would they use it (what is their motivation?), and where and when would the application be used?.

The total number of ideas generated by the 8 teams was 143. 38% of these ideas addressed aspects related to the functionality of the application under design. These ideas explored the space of the affordances the application would provide to its users. 28% of the scenario ideas related to the goals the users could have when interacting with the application under design. Exploring the potential users the application might have generated 14% of the ideas. 12% of the ideas the participants generated addressed the possible motivation for using the application (Table 2).

Problems	Scenario ideas						Total
	Users	Goals	Usage	Motivation	Location	Time	
Drawing	10	12	9	10	4	0	45
Text editing	1	4	13	1	0	0	19
Puzzle solving	5	9	13	1	1	3	32
Crosswords solving	4	15	20	5	2	1	47

Total	14%	28%	38%	12%	5%	3%	

Table 2 – Scenario ideas generated for each problem

The large number of ideas generated and their diversity supported the exploration of a broader number of design situations and possible good practices to tackle them. Scenario ideas related to functionality aspects addressed issues such as making all of the modifications on the shared resource visible to all the collaborators. Several notification strategies have been chosen as solutions throughout the workshops. Scenario ideas related to the goals of the users specified goals such as communicating with the other collaborators. As solution to that, several teams included in their scenarios the possibility of integrating an instant messaging tool in the application.

Results show that scenario-based design supports reflection in the context of design [3], creating opportunities for the designers' own reflection. This gives a degree of fluidity to the design process, able to support designers in their exploration. Moreover, having a set of initial questions they were asked to answer, the designers had a clear starting point for the exploration. This helped them initiate the design processes without feeling lost in identifying the focus points of the process.

The direct observation of the teams showed that the communication and interaction among the members of the teams was highly facilitated by the approach. They found scenarios to be an efficient technique for sharing their ideas and for making themselves understood within the team.

3.5 Patterns identified

Through the design workshops and based on the analysis of a set of existing software applications addressing the above described problems [13, 14, 19, 20, 21, 22, 23], a set of design patterns for the design of systems for synchronous collaboration was identified. The process followed for identifying the patterns is thoroughly described elsewhere [24], this paper focusing on the application of the scenario-based design in the mining process. The list of the briefly described patterns identified is presented in Table 2.

Design Pattern	Description
Who is the coordinator?	**Problem**. If more users work on the same resource in the same time, there needs to exist a coordination mechanism which: a). allows all collaborators to take part in the collaborations and b). maintains the resource in a consistent state at all times. The problem is how to determine who coordinates the collaborative process or what is the suited coordination mechanics for each concrete case. **Solution**. Possible coordination mechanisms: community coordination, one collaborator is the coordinator, locking, timers, a separate block of the shared resource for each collaborator.
Integrated chat	**Problem**. Collaborators should be able to exchange messages related to their collaboration, share knowledge based on each individual's expertise, and clarify any additional misunderstandings. **Solution**. Integrate an instant messaging feature in the design of the application.
Eyes wide open	**Problem**. Each collaborator must be able to visualize what the others are contributing to the process at any time. In addition to that, each contribution should be made visible to all the collaborators in real-time in order to allow participants' coordination. **Solution**. Update any changes on the commonly shared resource (drawing canvas, text area, puzzle/crosswords board) in the collaboration and notify (in real-time) all collaborators of these updates. The choice of notifications would depend on the context of the application.
Choose your	

collaborators	**Problem**. Users should be provided with the option of getting together and collaborating with their own peers. **Solution**. Allow each user to choose his/her collaborators as follows: a). Allow a user to search for his/her peers in the list of available users. b). Provide a list with all the users currently available. c). Allow users to invite each other to collaborate by creating a group. In case of games, one user may challenge others in joining a collaborative game. d). Allow users to join a group already created after s/he logs in to the application.
Collaboration, always social	**Problem**. Collaboration is, more than anything else, a social process. Hence, supporting collaborators through social features can only enhance and improve their collaborative process. **Solution**. Integrate mechanisms of tagging, ranking, annotating, and commenting in the application.
My contribution	**Problem**. Users should have available a straightforward and user friendly way to track their own contribution to the collaboration. **Solution**. Support each collaborator in tracking down his/her contribution to the collaboration, as follows: a). For the cases where the shared resource is textual (text editing, crosswords solving) and where the group of collaborators is relatively small, assign different colors to each collaborator. b). The applications for which the shared resource is an image may highlight one's contribution by representing (at one's request) only those shapes (in cases such as drawing) or pieces (in cases like puzzle solving) added by a particular user. c). Show tooltips containing information on the author of that particular part when a user drags the mouse over parts of the shared resources.
Track history of collaboration	**Problem**. Synchronous collaboration processes are being held in real-time, so it could be the case that a lot of the information on the dynamics of the collaborative group and on the knowledge exchanged is lost. **Solution**. Track the history of the collaboration and make it available either through repositories, log files or timelines.
With or without collaboration	**Problem**. Users might need, at times, to sketch their ideas before adding them to the area visible to all collaborators. Also, it might be the case that users need to try out solutions without interfering with the others' actions or without blocking the collaborative process. **Solution**. Provide users with an additional private area, not available to the other collaborators. Inside this area, each collaborator has total control and s/he is provided with tools specific to the context of the application.

Table 2 – Design patterns for the design of systems for synchronous collaboration

4. CONCLUSIONS

Design patterns are tools to support social creativity in that they allow communities of designers to make available knowledge related to design experiences, such as problems, solutions and design decisions. Identifying design patterns is a process known as design pattern mining. This paper describes one of the techniques used throughout a series of design workshops held for identifying design patterns in the design of software applications for synchronous collaboration.

Participants were asked to propose scenario ideas for the design of a collaborative application in one of the fields: drawing, text editing, puzzle solving, and crosswords solving. These scenario ideas were used for identifying commonly recurring design problems, solutions, and decisions in such design processes. The large number of ideas generated and their diversity supported the exploration of a larger number of design situations and possible good practices to tackle them.

Results show that scenario-based design support design pattern mining processes in that they allow designers to explore the design space of the application considered, identifying a broader set of possible issues to be faced in the design. Moreover, participants working in teams managed to communicate more efficiently.

ACKNOWLEDGEMENTS

Special thanks to all the participants in the workshops for the inspiring ideas and great designs. This work was supported by the *Initial Training Network* "Marie Curie Actions".

REFERENCES

[1] Borchers, J. *A Pattern Approach to Interaction Design*, 2001 (John Wiley & Sons, Inc.)

[2] Turner, S. G. *A Case Study Using Scenario-Based Design Tools and Techniques in the Formative Evaluation Stage of Instructional Design: Prototype Evaluation and Redesign of a Web-Enhanced Course Interface*, 1998, Ph.D. Thesis

[3] Carroll, J.M. Five Reasons for Scenario-Based Design. *Proceedings of the Hawaii International Conference on System Sciences* (HICSS '99), Vol. 3. IEEE Computer Society, Washington, DC, USA, 3051-.

[4] Fischer, G. Social creativity: turning barriers into opportunities for collaborative design, *Proc. of PDC'04*, Toronto, Canada, July 27 - 31, 2004, ACM, New York, NY, 152-161.

[5] Fischer, G., and Ostwald, J. Knowledge Communication In Design Communities, In R. Bromme, F. Hesse, & H. Spada (Eds.), *Barriers and Biases in Computer-Mediated Knowledge Communication*, 2005, Springer, New York, NY, pp 213 - 242.

[6] Díaz, P., Rosson, M. B., Aedo, I., and Carroll, J. M. Web Design Patterns: Investigating User Goals and Browsing Strategies, *Proc.of the Symposium on End-User Development*, Siegen, Germany, March 02 - 04, 2009, 186-204.

[7] Alexander, C. A *pattern language: Towns, buildings, construction*, 1977 (New York: Oxford University Press)

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

[9] Welie, M. *Patterns in interaction design*. Retrieved: June, 2^0th^ 2010. Available at: http://www.welie.com

[10] Crumlish, C., and Malone, E. *Designing Social Interfaces*, 2009 (O'Reilly Media, Inc.)

[11] Lukosch, S., and Schümmer, T. Communicating Design Knowledge with Groupware Technology Patterns: The Case of Shared Object Management. *CRIWG 2004*, 223-237.

[12] Schadewitz, N. Design Patterns for Cross-cultural Collaboration. *International Journal of Design*, 3(3), 2009.

[13] Amershi, S. and Morris, M. R. CoSearch: a system for co-located collaborative web search. *Proceeding of CHI '08*. ACM, New York, NY, USA, 1647-1656.

[14] Margaritis, M., Avouris, N., Kahrimanis, G. On Supporting Users' Reflection during Small Groups Synchronous Collaboration. *12th International Workshop on Groupware, CRIWG 2006*. LNCS 4154. Springer.

[15] Wenger, R. E. Cultivating Communities of Practice, 2002 (Harvard Business School Press, Boston)

[16] Murray, K.S. *Learning as Knowledge Integration*, 1995, Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-34902., University of Texas at Austin

[17] Fischer, G., and Ostwald, J. Knowledge Management--Problems, Promises, Realities and Challenges, *IEEE Intelligent Systems Journal, Special Issue "Knowledge Management: An Interdisciplinary Approach*, 2001, 60-72.

[18] Musen, M. A. Dimensions of knowledge sharing and reuse, *Comput. Biomed. Res.* 25(5), 1992, 435-467.

[19] Shah, C., Marchionini, G., Kelly, D. Learning design principles for a collaborative information seeking system. *Proceedings of CHI '09*. ACM, New York, NY, USA, 3419-3424.
[20] Adler, A., Nash, J. C., Noel, S. Evaluating and implementing a collaborative office document system. *Interact. Comput.* 18(4), 2006, 665-682.
[21] GoogleDocs http://www. *docs.google.com/*
[22] Klopfer, E., Perry, J., Squire, K, Jan, M. F., Steinkuehler, C. Mystery at the museum: a collaborative game for museum education. *In Proceedings of the 2005 conference on Computer support for collaborative learning: learning 2005: the next 10 years! (CSCL '05).* International Society of the Learning Sciences 316-320.
[23] Battocchi, A., Pianesi, F., Tomasini, D., Zancanaro, M., Esposito, G., Venuti, P., Ben Sasson, A., Gal, E., Weiss, P. L.. Collaborative Puzzle Game: a tabletop interactive game for fostering collaboration in children with Autism Spectrum Disorders (ASD). *Proceedings of the International Conference on Interactive Tabletops and Surfaces (ITS '09)*. ACM, New York, NY, USA, 197-204.
[24] Iacob, C. 2011. A Design Pattern Mining Method for Interaction Design. *ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS2011)*, Pisa, Italy, June 13-16, 2011

Contact: Claudia Iacob
University of Milan
Department of Computer Science
Via Comelico, 39-41
20135, Milan
Italy
Phone: +39 02 503 14007
Fax: +39 02 50314012
E-mail Address: iacob@dico.unimi.it

Claudia Iacob is currently a Ph.D. student at University of Milan, Department of Computer Science. She obtained her B.Sc. and her M.Sc. in Computer Science from the Department of Computer Science of Babes-Bolyai University, Cluj-Napoca. She is a Marie Curie early stage researcher and her research topics relate to interactive design patterns and tools for supporting creative design process through visual languages.