Integration of Usability Techniques into the Software Development Process

Xavier Ferre
Universidad Politecnica de Madrid
xavier@fi.upm.es

Abstract

Software development organisations are paying more and more attention to the usability of their software products. To raise the usability level of the software product, it is necessary to employ usability techniques, but their use is far from straightforward since they are not, in most cases, integrated with the software engineering development processes. Offering average software developers a way to integrate usability activities and techniques into their existing software development process can bridge the gap between usability and software engineering practice. The only requirement is for the existing process to be based on iterative refinement. We present a handy grouping of usability techniques as increments that developers can introduce into their software development process. We have arrived at this result by surveying the usability literature, adapting usability concepts to software engineering terminology, and examining the development time constraints on the application of usability activities and techniques.

1. Introduction

Usability is not addressed in software development as often as would be necessary to output highly usable software. It is properly addressed only in projects where there is an explicit interest in usability, and the quality of the system-user interaction is perceived as critical by the software development organisation. In this kind of projects, usability experts drive the development, using mostly usability-related techniques in the phases previous to coding. The processes followed and the techniques applied come from the HCI (Human-Computer Interaction) field, and they are not defined so as to be understandable in software engineering, so they cannot be employed "as they are" by average developers.

Larman states that there is probably no other kind of software development technique with a greater mismatch between its importance for the success of software development and the lack of rigorous attention and formal education than usability engineering and the design of the user interface [1]. Due to the increasing perception of usability as strategic for software development businesses, an increasing number of software development organisations are pursuing the aim of integrating usability practices into their software engineering processes, but it is not an easy endeavour [2]. Some proposals for integration ([3], [4]) present ad-hoc solutions that have been created for particular software development organisations. However, their approach is not general enough for them to be applied by other organisations.

One of the virtues of the HCI field lies in its multidisciplinary essence. This characteristic is, at the same time, the main obstacle to its integration with software engineering: while the HCI foundations come from the disciplines of psychology, sociology, industrial design, graphic design, and so forth; software engineers take a very different view, a typical engineering approach. Both fields speak a different language and they deal with software development from a different perspective [5]. We have tried to approach the integration of usability activities and techniques in a general software development process by employing the concepts and terminology that average software developers use, that is, software engineering terminology and concepts.

For this purpose, we first surveyed the usability literature to identify the characteristics that define a user-centred process and choose the usability techniques and activities best suited for inclusion in a software process. Then, we mapped all the findings in the usability field to the respective development activities, as expressed in software engineering. And, finally, we distilled the findings as process increments or deltas, which group similar kinds of usability techniques that are meant to be applied close together in terms of development time.

Each organisation can evaluate whether the type of process it has in place meets the minimum requirements for the incorporation of usability-oriented techniques and activities and, if the requirements are met, can add all or some of the process increments containing usability techniques to the existing software development process.

2. Characteristics of a User-Centred Software Development Process

We have studied the HCI literature to identify the characteristics that a software development process should have for it to be considered user-centred. [6], [7], [8], [9], [10] and [11] agree on considering iterative development as a must for a user-centred development process. The complexity of the human side in human-computer interaction makes it almost impossible to create a correct design at the first go. Cognitive, sociological, educational, physical and emotional issues may play an important role in any user-system interaction, and an iterative approach is the most sensible way to deal with these issues.

The other two characteristics that are mentioned by several sources are: active user involvement; and a proper understanding of user and task requirements. These two conditions can be met by introducing usability techniques that can help software developers with the integration of users into the design process and with the enhancement of requirements activities with specific usability aspects. On the contrary, the first condition (that is, to be based on iterative refinement) is an intrinsic characteristic of the software process, and it will be the only requirement for an existing development process to be a candidate for the introduction of usability techniques and activities.

When trying to output the activities that form part of a user-centred process, we found that the HCI field offers a heterogeneous landscape of methods and philosophies, like, for instance, usability engineering, usage-centred design, contextual inquiry, and participatory design. Each author attaches importance to a few techniques, and the terminology may vary from one author to another. For this reason, we first surveyed the HCI literature to identify the most agreed upon usability activities that should be part of the software development process. We have listed the usability activities in Figure 1, grouped according to the kind of development activity to which they belong.

We then surveyed usability techniques, which we allocated to the set of activities obtained previously. There are a host of usability techniques (we identified 82 techniques in our survey), some of which are just small variants of another technique. We have weighted the utility of each particular technique, trying to remove techniques whose objective can be attained by another technique from our selection. Where there were several candidate techniques, we considered the following criteria for our choice: how alien the technique is to software engineering, its general applicability, the cost of application, and its general acceptance in the HCI field. And we preferred to choose techniques that were less alien to software engineering, less costly to apply, and most generally accepted.

We have output a set of candidate usability techniques for inclusion in the software development process, where there is at least one technique to cover each usability activity and subactivity that can lead to an improvement in the usability level of the final software product. The set of candidate usability techniques selected amounts to a total of 51 techniques, which we are not detailing here for reasons of space.

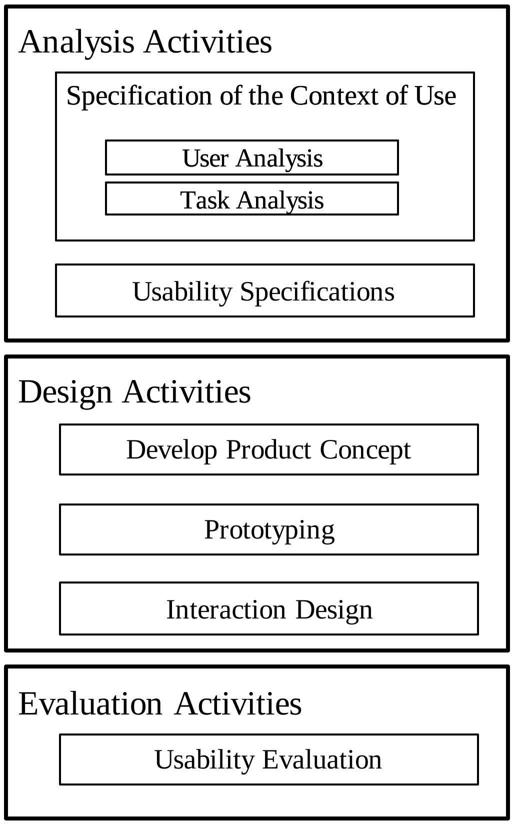

Figure 1. Usability activities grouped according to the generic type of development activity

3. Adaptation of Usability Activities to Software Engineering Development Process Concepts and Terminology

For the set of usability techniques we have chosen to be used by software developers, they need to be matched to the development process. Therefore, we need to adapt the results of our usability techniques and activities survey to software engineering concepts and terminology. Wherever possible, the SWEBOK [12] has been used as a basis for defining the activities in a traditional software development process.

Figure 2 shows the mapping between the usability activities from the usability literature (on the left-hand side) and a generic development process (on the right-hand side of the figure).

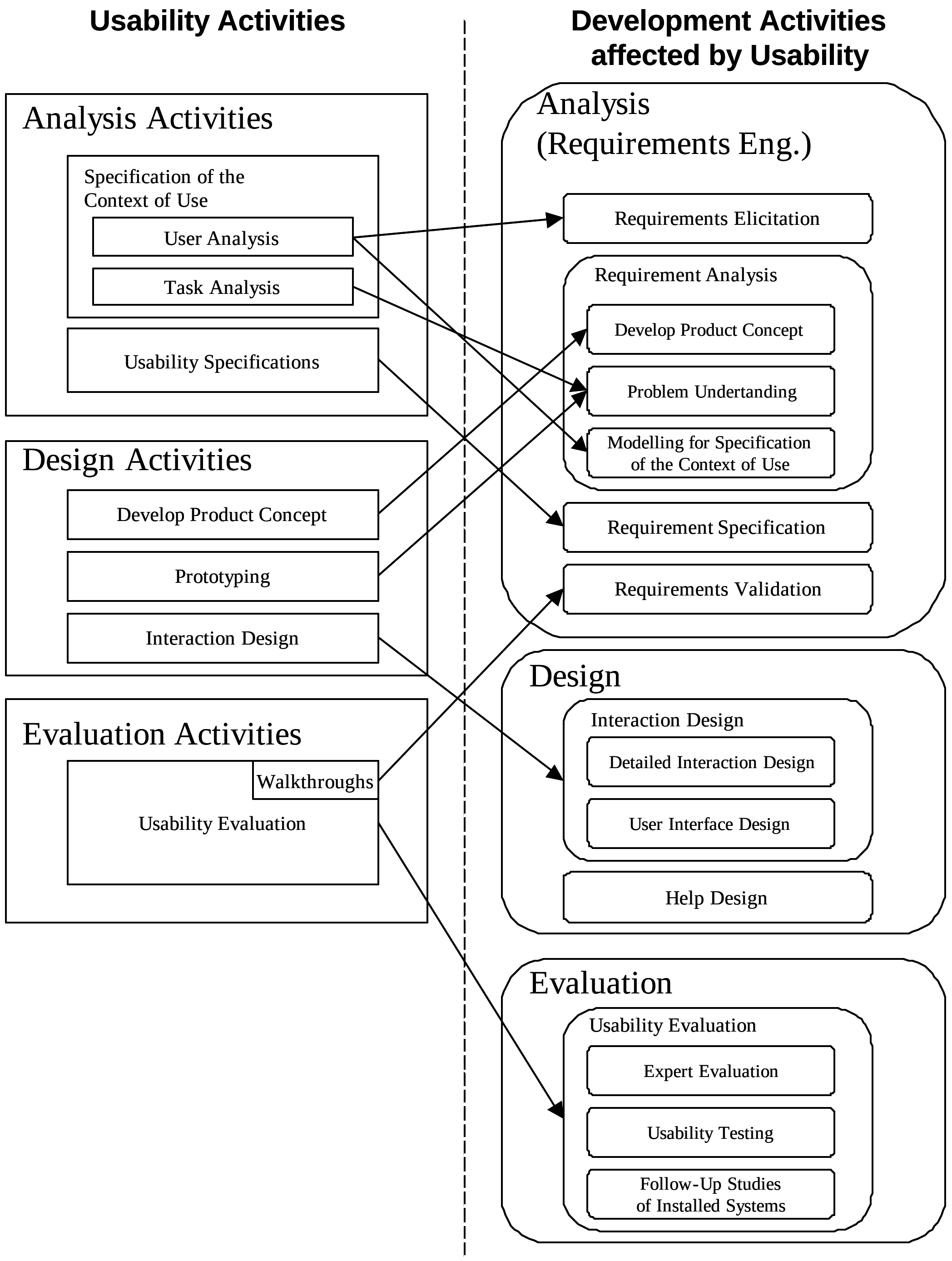

Figure 2. Mapping of usability activities to general development activities

Regarding analysis, we have taken the SWEBOK as a source for detailing the analysis activities. The relevant requirements engineering activities for our purpose are: requirements elicitation, requirement analysis, requirement specification and requirements validation. By relevant we mean that there are usability activities that interlink with development activities, so they can be mapped to certain software engineering analysis efforts. We have allocated two usability design activities to analysis, because of their close connection to requirements activities. They were considered as design activities in our survey, because they appear as such in usability literature. However, Prototyping is considered in software engineering as a technique that can be used for problem understanding, while Develop the Product Concept is the kind of design known as innovation design, and it is usually performed as part of requirements engineering efforts. The SWEBOK considers innovation design not as part of the software design activity, but as part of the requirements analysis activity. Therefore, we have included Prototyping and Develop the Product Concept as part of Requirement Analysis. Walkthroughs are a kind of evaluation activity that can be performed on analysis products, so it has been highlighted within Usability Evaluation in Figure 2 to be able to show its link with Requirements Validation.

On the other hand, we have not used the SWEBOK as a basis for design and evaluation activities. Unlike Analysis, we have usability activities in Design and Evaluation that are relatively independent of other design or evaluation development activities, and the structure for design and evaluation activities described in the SWEBOK is not suitable for usability activities.

We have divided design into Interaction Design and Help Design. The activity of Interaction Design is decomposed into Detailed Interaction Design and User Interface Design for the sake of clarity. Help Design was previously considered just a technique, since this was how it was presented in the usability literature. However, we have upgraded it to activity, since we have realised that the design of the help subsystem is an activity that is independent of the other design activities.

Usability Evaluation is also a separate activity from other development activities, but it is very complex. So we have subdivided it into the three main families of usability evaluation activities: Expert Evaluation, Usability Testing, and Follow-Up Studies of Installed Systems.

Having matched usability activities to software engineering activities, we could then map usability techniques to activities.

4. Allocation of Usability Techniques to Development Activities

Our primary basis for the allocation of usability techniques to development activities was the mapping of usability activities to general development activities.

We allocated all the usability techniques chosen in our survey to the development activities in Figure 2. For reasons of space, we cannot detail the full allocation here. However, Figure 3 shows the allocation for analysis

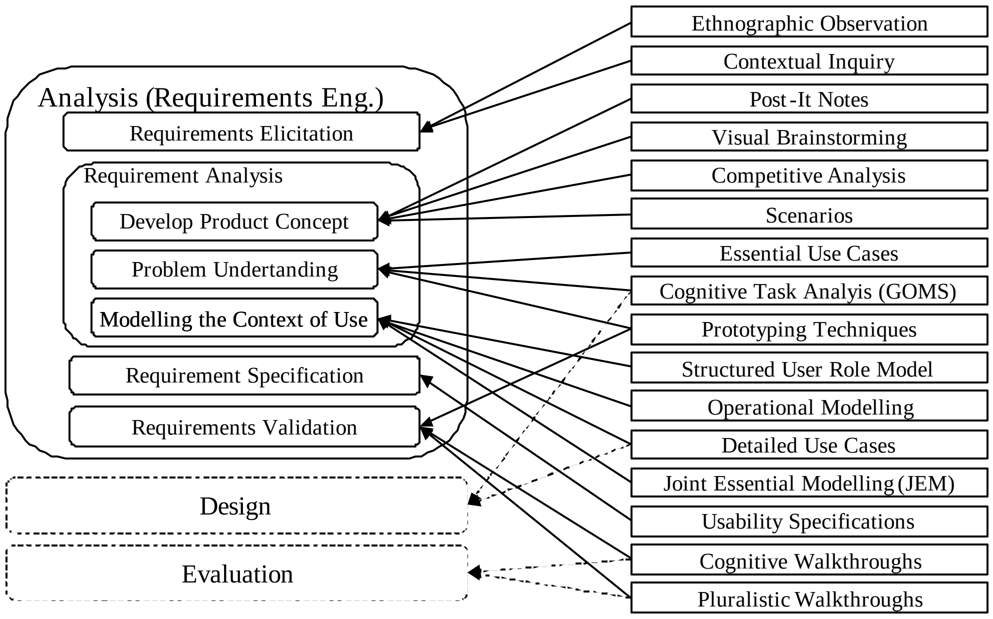

Figure 3. Allocation of usability techniques that apply in analysis

activities. The techniques are linked to the analysis activity to which they are allocated by an arrow. Some techniques may be applied in more than one activity, like Prototyping, which is used for Problem Understanding and also for Requirements Validation. Dashed lines mean that the technique is also applied during design or evaluation, like, for example, Detailed Use Cases, which are applied for Modelling the Context of Use, but also in design.

For the allocation of usability techniques to analysis activities, we have used their allocation to usability activities obtained from the literature survey, but we have also compared the objective of each technique and its products with the definition of analysis activities given in the SWEBOK.

5. Time Constraints for the Application of Usability Activities and Techniques

It is not enough just to allocate usability techniques to development activities, since not all usability techniques are applicable at any time in an iterative development. Any iterative process is divided into stages and, while not all iterative processes are the same, they usually follow a similar pattern regarding development time. We have defined a generic set of development stages that is applicable to most iterative processes. This model of process stages is shown in Figure 4. Prior to the iterative cycles, there is an initial exploration stage, which we have called Elaboration.

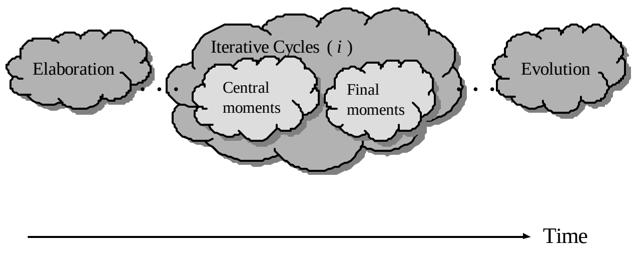

Figure 4. Stages in the development process

Afterwards, within the iterative cycles, we make a distinction between the main part of each cycle (Central moments) and the last part of each cycle (Final moments), where certain activities are performed, typically evaluation activities. Finally, when the system has been installed and is operational at the customer’ s site, the cycles are called Evolution.

We have studied the time of application of the different usability techniques in each activity to output a distribution of work across the different kinds of activities, related to the time in the development process when each effort is performed, as shown in Figure 5. The X-axis

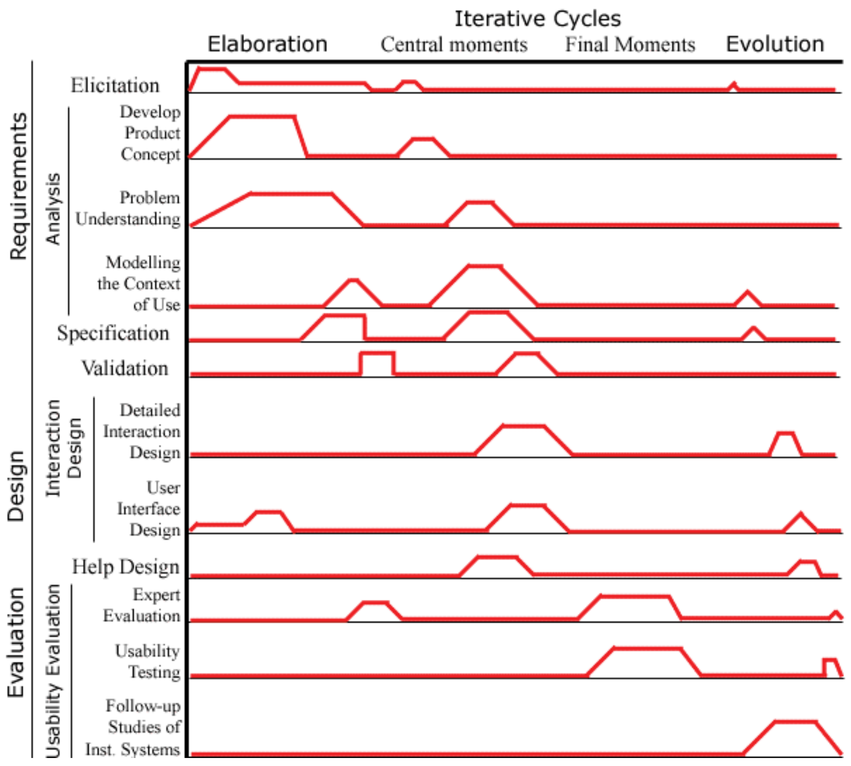

Figure 5. Amount of work on each activity at development stages

represents time. Therefore, the slopes in the lines denote some precedence between the different kinds of activities, like, for example, between the different requirements activities: first, there is some elicitation, followed by some development of the product concept (overlapping with the previous task), and then some problem understanding activities, and so on. Note that the amount of work on each activity is approximate, it should not be taken literally.

We focused first on usability techniques to allocate activities to moments in development time. We allocated each usability technique to a development stage according to its time of application in a user-centred development process. For example, techniques for developing the product concept are aimed at the very first development effort, where the needs are identified and the general system scheme is established, that is, the Elaboration stage.

Considering the time constraints for the techniques applied in each usability activity, we have then identified time constraints for usability activities. For example, elicitation is mostly performed in the Elaboration cycles (with more emphasis on the early stages), while some elicitation activities are performed at the beginning of the central moments within the Iterative Cycles, and a small amount of work may be done in Evolution cycles.

6. Definition of Process Increments

As a result of the work described above, we have classed the usability activities and techniques to be applied in the development process as increments, which we have called deltas, grouping techniques that are meant to be applied together according to the nature of the activities to which they belong (analysis, design or evaluation), and to the moment in development time when they can more effectively improve the usability of the software product.

We have defined seven deltas in order to get a better match with the general stages of an iterative software development process:

- **Δ1**: Early Analysis
- **Δ2**: Usability Specifications
- **Δ3**: Early Usability Evaluation
- **Δ4**: Regular Analysis
- **Δ5**: Interaction Design
- **Δ6**: Regular Usability Evaluation
- **Δ7**: Usability Evaluation of Installed Systems

Analysis activities are the ones that allow for a greater subdivision and call for careful integration with software engineering activities. Therefore, we have three deltas for analysis activities (Δ1, Δ2 and Δ4) plus Δ3, which, although formed by evaluation techniques, is applied in analysis activities. Traditional usability design activities are quite uniform and can be integrated in just one delta, Δ5. Usability evaluation activities, apart from the above-mentioned Δ3, have been divided into the activities to be performed during the main iterative cycles (Δ6), and the activities to be performed once we have an operational system working in the customer organisation (Δ7).

Figure 6 shows how the deltas group similar kinds of activities to be performed close together in development time. Each triangle represents one of the deltas, and they are placed over the distribution of work represented in Figure 5. The location along the X-axis represents the moment in development time when the delta should be applied, and the location on the Y-axis represents the kind of activities the delta groups. Note that the size of the deltas is not meaningful, as its only purpose is to cover the activities each increment contains.

Each process increment or delta is described according to the following structure:

- **Purpose**: The reasons why the delta should be added to an existing development process in order to improve the usability level of the resulting software product.
- **Phase**: Main type of activity: analysis / design / evaluation
- **Stage**: Development process stage where it is applicable.
- **Participants**: Members of the development team and other stakeholders who are meant to participate in the application of the techniques.
- **Activities/Techniques/Products**: List of the usability techniques that the delta groups, along with the documents or models produced by each technique. The techniques are grouped by the activity required to produce each product.

To make deltas handy for developers, each delta is summarised on just one page. Again, for reasons of space, we cannot detail all the deltas here. By way of an example, Table 1 gives a description of Δ1: Early Analysis. The set of techniques that form the delta are organised according to the activity to which they are allocated. This means that developers can add them more easily to their development process.

The products to be produced or refined by the application of each technique are detailed as well. For example, Ethnographic Observation and Contextual Inquiry are elicitation techniques and they help to build the Structured User Role Model, the Operational Model and the Use Case Diagram. Note that the same product may be related to several techniques, like, for example, the User Structured Model. This means that this product is defined by applying a conjunction of the usability techniques that produce or refine it.

Along with the description of the deltas, we offer developers a catalogue of usability techniques. The catalogue contains a brief description of the usability techniques mentioned in the increments (it includes 31 usability techniques). Deltas make reference to usability techniques that are mostly unknown to average developers, so the catalogue aims to provide them with an idea of what each technique is about. The catalogue refers developers to HCI literature and usability training for a deeper understanding of the techniques.

7. Conclusions

One of the reasons why usability techniques are not regularly used in software development, despite how critical software usability is for the overall quality of the software product, is the lack of integration with software engineering concepts, terminology and process.

We have presented a way of integrating usability activities and techniques into the software development process. The integration of usability techniques and activities is packaged in the form of increments that are to be incorporated at different places in a software development process. The appeal for a software development organisation lies in the fact that it does not have to abandon the in-house process to adopt some improvements, as it is enough to just modify the existing process by adding some or all of the proposed increments.

Offering average software developers an organised and manageable set of usability increments for their software development process, then, supports the current demand for flexible software processes.

We are currently applying the results we have presented here to three real projects at two companies. Through this validation, we intend to not only check and improve our proposal, but also to study the minimum support and training necessary for developers who are new to usability, for them to routinely apply the proposed increments and their associated techniques.

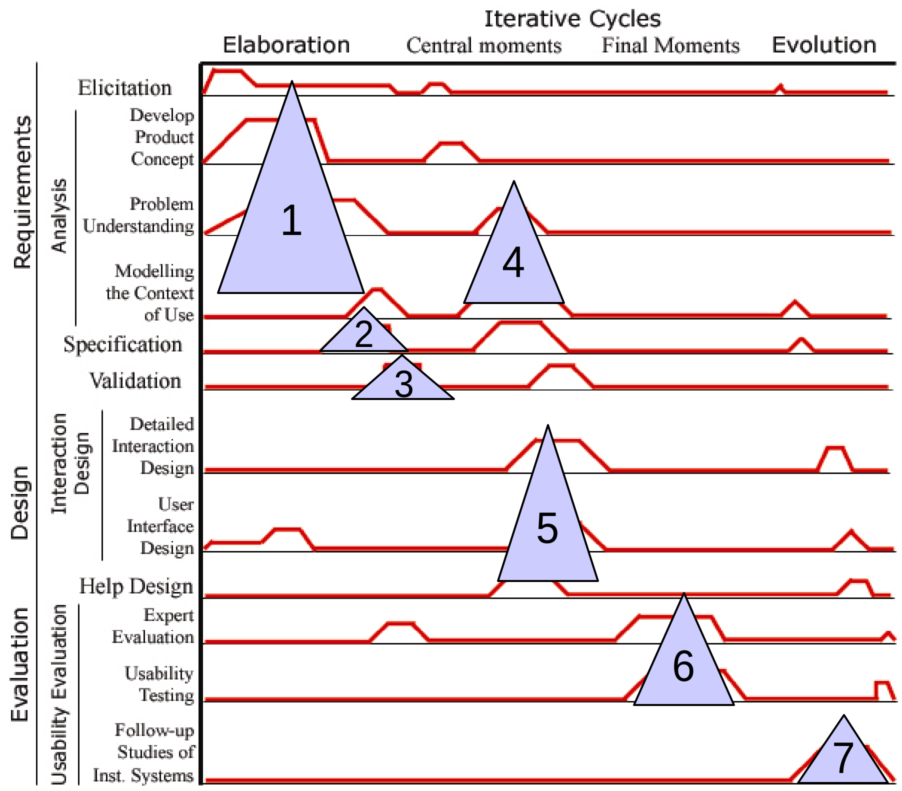

Figure 6. Grouping of usability activities in deltas according to the moment of application in development

Acknowledgments

This research has been funded by the “Ministerio de Ciencia y Tecnología” of Spain, project number TIC2002-00320.

Table 1. Δ1: Early Analysis

PURPOSE	Usability offers several techniques for analysis at the early stages of the project. These activities can give the tasks of requirements elicitation and analysis the user-centred flavour that ensures that usability is sufficiently catered for in later development activities.	
PHASE	Analysis	
STAGE	Elaboration	
PARTICIPANTS	Customer, users, developers	
ACTIVITIES	**TECHNIQUES**	**PRODUCTS**
ELICITATION	Ethnographic Observation	*-Structured User Role Model* *-Operational Model* *-Use Case Diagram*
	Contextual Inquiry	
REQ. ANALYSIS - MODELLING THE CONTEXT OF USE	Structured User Role Model	*-Structured User Role Model*
	JEM	*-Structured User Role Model* *-Essential Use Cases* *-Use Case Diagram*
	Operational Modelling	*-Operational Model*
REQ. ANALYSIS – DEVELOP PRODUCT CONCEPT	Post-It Notes	*-Product Concept*
	Visual Brainstorming	
	Competitive Analysis	*-Product Concept* *-List of needs and key/differentiating features*
	Scenarios	*-Scenarios*
REQ. ANALYSIS – PROBLEM UNDERSTANDING	Essential Use Cases	*-Essential Use Cases*
	Prototypes (paper and chauffeured)	*-Paper prototype*

8. References

[1] C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process (2nd Edition)*, Prentice Hall PTR, Upple Saddle River, NJ, 2002.
[2] IEEE, Special Issue on “Usability Engineering in Software Development”, *IEEE Software*, Guest Editors: N. Juristo, H. Windl, and L. Constantine, Vol. 18, no. 1, January/February 2001.
[3] J. Anderson, F.Fleek, K. Garrity, and F. Drake. “IntegratingUsability Techniques into Software Development”. *IEEE Software*, vol.18, no.1, January/February 2001, pp. 46-53.
[4] K. Radle, and S. Young. “Partnering Usability with Development: How Three Organizations Succeeded”. *IEEE Software*, vol.18, no.1, January/February 2001, pp. 38-45.
[5] X. Ferre, N. Juristo, H. Windl, and L. Constantine. “Usability Basics for Software Developers”, *IEEE Software*, Vol. 18, no. 1, January/February 2001, pp. 22-29.
[6] J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, and T. Carey, *Human-Computer Interaction*. Addison Wesley, Harlow, England, 1994.
[7] ISO, *International Standard ISO 13407. Human-Centred Design Processes for Interactive Systems*, ISO, Geneva, Switzerland, 1999.
[8] L. L. Constantine, and L. A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. Addison-Wesley, New York, NY, 1999.
[9] D. Hix, and H.R. Hartson. *Developing User Interfaces: Ensuring Usability Through Product and Process*. John Wiley and Sons, New York, NY, 1993.
[10] B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Reading, MA, 1998.
[11] J. Nielsen. *Usability Engineering*. AP Professional, Boston, MA, 1993.
[12] IEEE Software Engineering Coordinating Committee. *Guide to the Software Engineering Body of Knowledge - Trial Version 1.00*. IEEE Computer Society, Los Alamitos, California, May 2001.