ISSN 2281-4299

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

HORIZON: A Development Methodology for Collaborative Projects

Tiziana Catarci
Shah Rukh Humayoun
Francesco Leotta
Andrea Marrella
Massimo Mecella
Antonella Poggi

Technical Report n. 2, 2015

HORIZON: A Development Methodology for Collaborative Projects

Tiziana Catarci
Sapienza Università di Roma, Italy
catarci@dis.uniroma1.it

Shah Rukh Humayoun
Technische Universität Kaiserslautern
humayoun@cs.uni-kl.de

Stephen Kimani
Jomo Kenyatta University, Kenya
stephenkimani@googlemail.com

Francesco Leotta
Sapienza Università di Roma, Italy
leotta@dis.uniroma1.it

Andrea Marrella
Sapienza Università di Roma, Italy
marrella@dis.uniroma1.it

Massimo Mecella
Sapienza Università di Roma, Italy
mecella@dis.uniroma1.it

Antonella Poggi
Sapienza Università di Roma, Italy
poggi@dis.uniroma1.it

Abstract
In this paper, we introduce a methodology to be employed for collaborative projects. This methodology, called HORIZON, has been successfully employed in the past for FP6 and FP7 EU projects, but we strongly believe it can be fruitfully employed for H2020 projects as well, guaranteeing quality requirements to be met.

Keywords
Collaborative project ; development methodology ; EU project ; user-centered design ; agile methods

1. INTRODUCTION
The methodologies and technologies to be developed in the context of a European project are radically different from the existing solutions and there is a need for a notable amount of experimentation. Thus, the solutions developed within the project need to be interactively explored with the industrial users and the new ideas and concepts need to be tested. The users will be able to offer concrete requirements, but, due to the innovative approaches proposed, further requirements need to be explored with the use of prototypes as a requirements elicitation tool. Prototypes will therefore be used to present ideas and alternative approaches and to get rapid feedbacks. A standard process life cycle is therefore counterproductive, and for this, there is the need to work with an innovative and creative model that supports the iterative and incremental evolution with rapid development pieces, which exploits the users for getting continuously requirements and feedbacks for designing and developing the outcome systems more interactive and more effective.

This paper introduces a best practice that encompasses the integration of three highly usable and effective approaches that are: the spiral life cycle model, the agile development method, and the use of user-centered design (UCD) techniques. We have experienced using part/parts of this model successfully in many FP6 and FP7 European research projects (i.e., WORKPAD[1] [HCdL+09] and SM4ALL22[2]) but we strongly believe it can fruitfully serves for H2020 framework projects as well.

In the following we will refer to the fictitious HORIZON project in order to introduce our methodology.

2. BACKGROUND

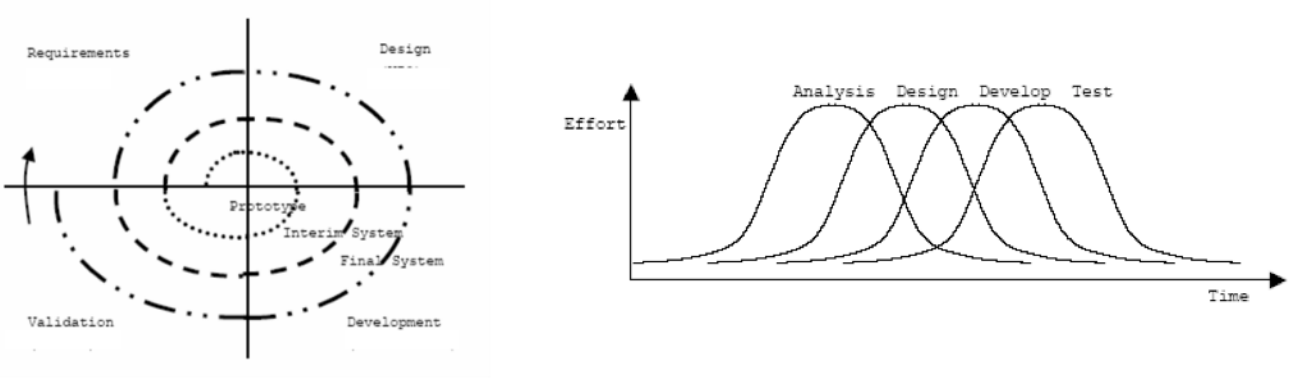
2.1 THE SPYRAL LIFE CYCLE MODEL
The spiral life-cycle [Boe88] is iterative, starting with initial requirements gathering followed by rapid proposition on new basic research ideas, design, and development of a prototype implementing them, that is then tested and validated by the users and is enhanced in the next iteration with users feedbacks and additional requirements. The test results provide a new set of requirements, through redefining or completing the old set. With the second set of requirements a new design and development is performed. After that, users are required to test the system again.

That process (Figure 1(a)) is continued till a complete set of requirements that meets all user needs has been implemented. The diagram in Figure 1(b) outlines the overlap between the four phases of the spiral with respect to concurrency and level of effort.

2.2 USER-CENTERED DESIGN METHODOLOGY
The Consortium has to pay special attention to the involvement of the industrial users in the process of research, development and prototype creation and introduction. The HORIZON prototype will be developed by using a user-centered and participatory methodology. User-centered design methodology places the end-users (as opposed to the "thing") at the center and involves the users in design and development activities [Dix04, GGB+03, Blo06, PRS02]. It is a design philosophy that focuses also on cognitive factors (such as perception, memory, learning, problem-solving, etc.) as they come into play during people interactions with things and/or systems. It seeks to answer questions about users and their tasks and goals, and further on uses the findings to

[1] EU project WORKPAD: http://www.dis.uniroma1.it/~workpad

[2] EU project SM4ALL: http://www.sm4all-project.eu/



(a) Spiral life cycle model (b) Intensity and overlap of effort during the spiral development cycle

Figure 1. Spiral Life Cycle Model [Boeh88]

drive development and design. This is done, for example, by representing or modelling users in scenarios and personas, having users test prototypes (either paper or working prototype), and involving users in design decisions (e.g., thorough participatory design), etc. The activities in UCD methodology aim at reducing the risks of the software project and increasing the overall product quality [VMSC02, Dix04, PRS02]. User-centered design is increasingly seen as essential for the creation of successful products and prototyping Information and Communication Technologies (ICT) applications; it enables understanding the needs of users early in the design and development process, shaping product design to their needs, and validating the acceptability and usability of the ICT product or prototype application, thus prevent product failure at the end [AFGD01, JWC01, RY01]. There is an international standard that is the basis for many user-centered design methodologies. This standard (ISO 1340723[3]: Human-Centered design process) provides guidance on user-oriented design process for including human-centered activities throughout a development life-cycle, but does not specify specific techniques to be used during such a development. Few of the methods in UCD are: scenarios, focus groups, interviews, card sort, contextual inquiry, story board, task analysis, prototype evaluation methods (heuristic, thinking aloud, cooperative, controlled experiments, etc.), log file analysis, survey, etc. Each of them supports some particular type of activities and suits for specific purposes. A proposal for the techniques and methods to be used has been specified in detail by Sapienza and tested in the WORKPAD project [HCdL+09].

2.3 AGILE DEVELOPMENT APPROACH

A software development approach that has been emerging in the last decade is the agile approach that is used for constructing software products in iterative and incremental manner: where each iteration produces working artifacts that are valuable to the customers, which also includes the industrial users, and to the project. This is performed in a highly collaborative fashion in order to produce quality products that meet the requirements in cost effective and timely manner. Agile development broadly came to focus when defined by Agile Alliance in the Agile Manifesto[4] in 2001 with twelve principles and four key values as: Individuals and interactions over processes and tools; Working software over comprehensive documentation; Customer collaboration over contract negotiation; Responding to change over following a plan.

There exist a number of methods, which share these principles and key values under the umbrella of agile development approach. Each methodology differs slightly from the others, depends on different factors, but all are lightweight with short-term iterations in incremental fashion and they involve customers to establish, prioritize, and verifying the requirements throughout the whole development lifecycle. Extreme Programming (XP)[5], SCRUM[6], Agile Unified Process (AUP)[7], Feature Driven Development (FDD)[8], Agile Modelling[9], Adaptive Software Development (ADP), Crystal Clear and Other Crystal Methodologies, Dynamic System Development Method (DSDM)[10] are few well-known agile methods [ASRW02]. Field surveys[11] in companies, which work with agile development, show the overall improvement in nearly all areas of product development. Agile software teams are mostly small self-organizing cross-functional teams, which work by breaking tasks into small increments with minimal planning. Iterations are normally short time frames, called as time boxes, which typically last from one to four weeks [ASRW02]. In each

[3] ISO/DIS 13407: Human Centered Design for Interactive Systems, ISO, 1999

[4] Agile Alliance 2001. Manifesto for Agile Software Development - http://www.agilealliance.org.

[5] http://www.extremeprogramming.org/

[6] http://scrummethodology.com/

[7] http://www.ambysoft.com/unifiedprocess/agileUP.html

[8] http://www.featuredrivendevelopment.com/

[9] http://www.agilemodeling.com/

[10] http://www.dsdm.org/

[11] VersionOne, Inc.: 2nd Annual Survey: The State of Agile Development. 2007 - http://www.versionone.com/pdf/StateOfAgileDevelopmet2_FullDataReport.pdf

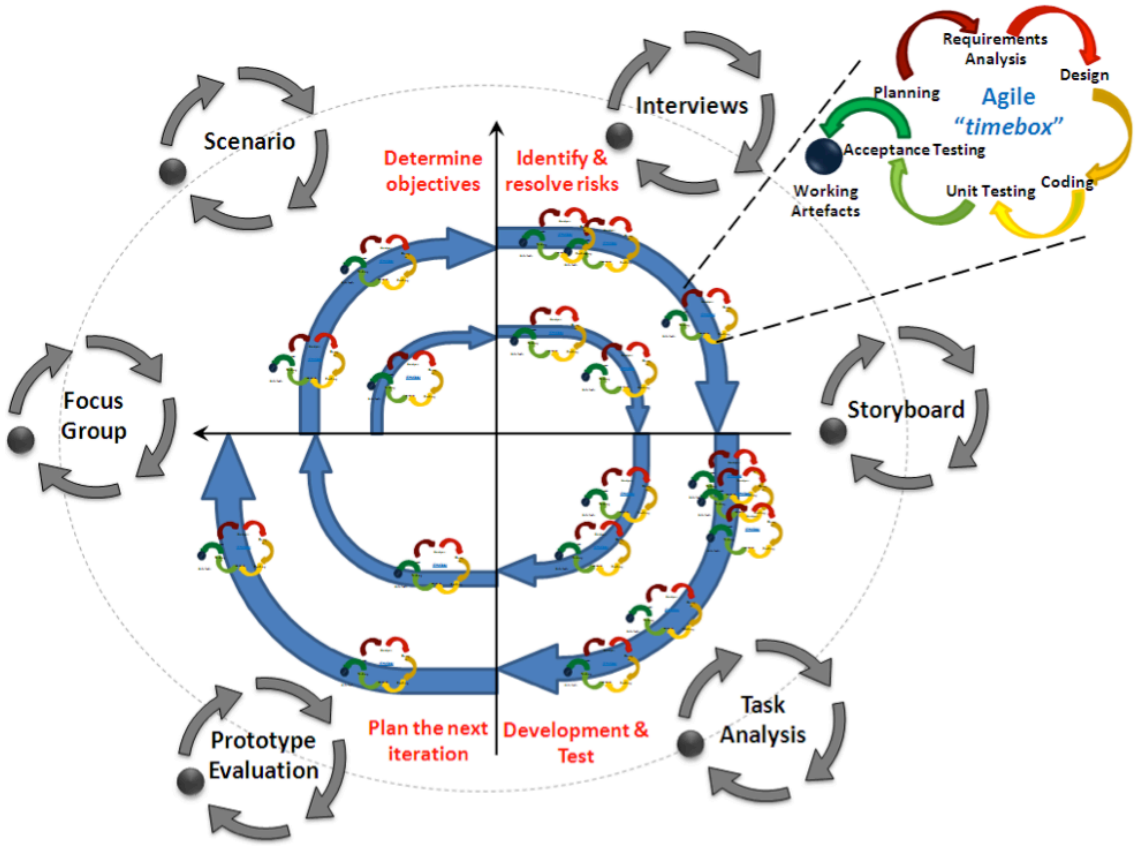


Figure 2. HORIZON life-cycle methodology

iteration software teams work through a full software development cycle including planning, requirement analysis, design, coding, unit testing, and additional acceptance testing for that iteration when a working product is demonstrated to stakeholders. The effect of this process is the reduction of overall risks and lets the project adapt to changes quickly and reduce the time of development. Although, it is possible that a single iteration may not add enough functionalities to make it for market release, but the goal is to have an available release with minimum bugs at the end of each iteration, so a release can take multiple iterations.

3. PROPOSED APPROACH

HORIZON will provide a set of new tools and technologies that will work very differently from the existing solutions, so there is the need for an innovative and creative life-cycle methodology that supports the iterative and incremental evolution with rapid deployment for getting early feedbacks from users to make these technologies and tools more effective, efficient, and interactive.

For this, the proposed life-cycle methodology integrates the above described approaches and uses them in three directions for getting effective outcomes:

- the spiral life-cycle model to manage the whole life-cycle of the project in iterative and incremental manner with focus on prototyping technique;
- the use of agile development cycles in each cycle of the spiral model for developing tools and technologies for rapid deployment in order to get early users feedbacks and to improve productivity in cost effective and timely manner;
- and the continuous deployment of the user-centered design techniques throughout the whole life-cycles (within both spiral and agile development life-cycles) for involving the users throughout the process of design and development, for checking and evaluating the developed tools and technologies early to reduce the risks of failure of the project.

The prototyping approach of the spiral model is suitable, as users will be able to check the system in the early stages. From each prototype, the users will be able to assure the capabilities of the system, and the following iterations will let them confirm that their needs have been met. This will allow a periodic re-validation of the project development by the users groups. In each iteration of the cycle, the users' acceptance will be required and partial results will be disseminated among all of them.

HORIZON generally has to produce a set of tools and technologies within the initial 18 months of the project, largely based on existing technologies and artefacts, in order to act as technology probes for requirements elicitation. The agile development approach is well suited to get results and deployment early. As each targeted tool or technology will need a different time and effort to build, we will plan multiple agile development cycles for building each tool and technology so to increase its functionalities in later cycles. Due to the use of short-time nature of agile cycles, it will be easy to test and evaluate those tools early with the users and this will also help to improve and complete the requirements for the coming

iterations. So, instead of waiting for the implementation at the end of a spiral model, the use of the agile development approach in this manner will give the advantage to test and evaluate those tools and technologies many times in short-terms iterations, and will provide a solid product to test and evaluate on larger scale at the end of a spiral cycle. The users of HORIZON will have a significant impact on the development and creation process and are considered to be the most important drivers of the innovation within the project. They will be actively involved in all stages of the application creation. The iterative spiral model of development also matches with user-centered approaches to software design where iterative prototyping is regarded as essential.

In particular, the use of technology probes [HMW+03] has proved invaluable in eliciting users requirements for radically new products. Technology probes are indeed a very early prototype, which may use "lash-up" technology, or in some way be incomplete or exploratory, yet is sufficiently robust to be usable in a real situation. The aim of the probe is to enable users to envisage novel technology and imagine how it can fit into their factories.

For designing, developing, and deploying these technologies rapidly, HORIZON will use agile development cycles within each cycle of the spiral model. Although the UCD and agile development are different approaches that were raised within different disciplines, the basic set of concepts and philosophy has no fundamental contradictions [Blo05]. Thus, by emphasizing the shared ideas that go hand in hand, these two approaches can be integrated to develop high quality and usable software products. Even the small investment of UCD activities in agile development gives the benefits in large [Det07]. Many recent studies, ours as well as others, have shown the integration and usage of UCD and usability evaluation techniques in agile and other software development processes at different levels with different granularities as well as tools for managing these UCD activities during development times [AFGD01, Fer03, Hud03, HLRS04, Blo05, DCHK07, DHC08, HDC09].

Figure 2 shows the HORIZON life-cycle methodology in a nutshell. As described early, the overall project will be managed through the spiral model and the main central incremental ring in the diagram represents the cycles of the spiral model. The small rings attached to the spiral ring represent the agile development cycles. These agile development cycles are for managing the development of tools and technologies that HORIZON will produce during the project life-cycle. The reason for managing them through agile cycles is because it is best suited for short-term iterations and for developing small products; whereas, on the other side, the spiral model is considered best suited for handling big projects and strategically combines elements of both design and prototyping, in order to get advantage from both top-down and bottom-up approach. Thus, the integration of managing the whole project through the spiral model and handling the development of tools inside the project through agile development cycles gives a framework for getting the best results in cost effective and timely manner and reduces the risks of the failure of the project. The upper top right side of the diagram presents a complete life-cycle of agile development, and each one of the agile cycle attached to the spiral cycle represents this complete life-cycle, from planning to producing working artefacts. The number of agile cycles attached to a part of the spiral cycle totally depend on the status of the project and the tools in progress. There may be, in one moment, a number of agile cycles running in parallel as there may be, at the same time, small teams working on different tools inside the project. The attachment of agile cycles seams more congested towards the second and third part of the spiral cycle due to the nature of these parts. The outer soft-line ring that covers the spiral model cycles represents the use of UCD methods throughout the project life-cycle. This covering of the whole spiral model emphasizes the idea that UCD methods are involved actively in both kinds of approach, the spiral and the agile one. Each UCD method has its own ring that represents the continuous evolving and refining of the results.

Note that while HORIZON will adopt a user-centered approach, this is not in order to create the most well-tuned product, but instead to *(a)* produce software of sufficient usability that it can be used by large numbers of users during experimental deployment, and, most importantly, to *(b)* uncover broad and fundamental knowledge about the potential and actual use of this novel class of systems.

4. CONCLUSIONS

In this paper we presented a practical methodology to manage a collaborative European projects where a set of artifacts are build aiming at meeting a specific set of qualitative and functional requirements.

5. REFERENCES

[AFGD01] J. Anderson, F. Fleak, K. Garrity, F. Drake: Integrating usability techniques into software development. Software, IEEE, Vol. 18, No. 1, pp. 46–53, 2001.

[ASRW02] P. Abrahamsson, O. Salo, J. Ronkainen, J. Warsta: Agile software development methods: Review and analysis, 2002.

[Blo05] S. Blomkvist: Towards a model for bridging agile development and user-centered design. Human-Centered Software Engineering-Integrating Usability in the Software Development Lifecycle, pp. 219– 244, 2005.

[Blo06] S. Blomkvist: User-Centred Design and Agile Development of IT Systems. Licentiate thesis, Department of Information Technology, Uppsala University, December 2006.

[Boe88] B.W. Boehm: A spiral model of software development and enhancement. Computer, Vol. 21, No. 5, pp. 61–72, 1988.

[DCHK07] Y. Dubinsky, T. Catarci, S. R. Humayoun, S. Kimani: Integrating user evaluation into software development environments. 2nd DELOS conference on digital libraries, Pisa, Italy, 2007.

[Det07] M. Detweiler: Managing UCD within agile projects. ACM interactions, Vol. 14, No. 3, pp. 40–42, 2007.

[DHC08] Y. Dubinsky, S. R. Humayoun, T. Catarci: Eclipse plug-in to manage user centered design. Workshop on the Interplay between Usability Evaluation and Software Development (I-USED), Pisa, Italy, 2008.

[Dix04] A. Dix: Human-computer interaction. Prentice hall, 2004.

[Fer03] X. Ferre: Integration of usability techniques into the software development process. International Conference on Software Engineering (Bridging the

gaps between software engineering and human-computer interaction), pp. 28–35, 2003.

[GGB+03] J. Gulliksen, B. Goransson, I. Boivie, S. Blomkvist, J. Persson, A. Cajander: Key principles for user-centred systems design. Behaviour and Information Technology, Vol. 22, No. 6, pp. 397–409, 2003.

[HCdL+09] S.R. Humayoun, T. Catarci, M. de Leoni, A. Marrella, M. Mecella, M. Bortenschlager, R. Steinmann: Designing mobile systems in highly dynamic scenarios: The WORKPAD methodology. Knowledge, Technology & Policy, Vol. 22, No. 1, pp. 25–43, 2009.

[HDC09] S.R. Humayoun, Y. Dubinsky, T. Catarci: Ueman: A tool to manage user evaluation in development environments. Software Engineering, 2009. ICSE 2009, pp. 551–554, 2009. [HLRS04] B. Hwong, D. Laurance, A. Rudorfer, X. Song: User-centered design and agile software development processes. CHI, April, pp. 25–29, 2004.

[HMW+03] H. Hutchinson, W. Mackay, B. Westerlund, B.B. Bederson, A. Druin, C. Plaisant, M. Beaudouin-Lafon, S. Conversy, H. Evans, H. Hansen, et al: Technology probes: inspiring design for and with families. SIGCHI conference on Human factors in computing systems, pp. 17–24, 2003.

[Hud03] W. Hudson: Adopting user-centered design within an agile process: A conversation. Cutter IT Journal, Vol. 16, No. 10, pp. 5–12, 2003.

[JWC01] N. Juristo, H. Windl, L. Constantine: Guest editors' introduction: Introducing usability. IEEE Software, Vol. 18, No. 1, pp. 20–21, 2001.

[PRS02] J. Preece, Y. Rogers, H. Sharp: Interaction design: beyond human-computer interaction. Hoboken (NJ): John Wiley & Sons, 2002.

[RY01] K. Radle, S. Young. Partnering usability with development: How three organizations succeeded. Software, IEEE, Vol. 18, No. 1, pp. 38–45, 2001.

[VMSC02] K. Vredenburg, J.Y. Mao, P.W. Smith, T. Carey: A survey of user-centered design practice. SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves, pp. 471–478, 20