Agile method in the support of UI Context-Aware Adaptation

Nesrine Mezhoudi, Jorge Luis Perez Medina and Jean Vanderdonckt
Université catholique de Louvain –
Louvain School of Management - Lilab
Louvain-la-Neuve, Belgium
E-mail: {nesrine.mezhoudi, jorge.perezmedina, jean.vanderdonckt}@uclouvain.be

***Abstract*—Although the adaptation approaches are evolving with changes in the technical landscape, their purposes are still to increase user satisfaction and result in successful interactions. Commonly, adaptation is intended to ensure context-aware interaction meeting user expectations. Thereby 'context-awareness' as well as 'user-centeredness' becomes mandatory to adapt the UI in response to context changes. However, interface adaptations are mostly managed at design time, instead of conforming current situations and the ambient-contexts. Thus, an accurate adaptation approach should be context-aware, flexible, incremental and have a crosscutting impact on software patterning, with an insignificant cost. In order to address these main shortcomings and support stakeholders to bridge the gap between adaptation goals and user needs, this proposal conveys a theoretical framework establishing runtime context-aware adaptation within an agile perspective.**

Keywords: framework; runtime adaptation; context-awareness; Agility.

I. INTRODUCTION

Adaptation is recognized as the key factor for the success of an interaction by the HCI community [10, 13]. Generally, adaptation concerns three main concepts establishing the context of use: the user, the platform and the environment. Adaptation is aimed at accommodating context requirements and users preferences in order to improve the interaction.

Although there were successful adaptive systems 10 years ago, they did not often consider varying context information during execution. Given the changing status of user needs and expectations, adapting UIs often demands complex inferences and strategies for acquiring and considering up-to-date contextual facts. Likewise, adaptation should have a crosscutting impact on the software design and appearance depending on interaction features and the ambient-context with an insignificant cost [28]. By attempting to cut with earlier interfaces that often needed recompilation for upgrades, which incurred increased cost, delay, and risk, UIs shift to a runtime adaptation paradigm. User interfaces turn out to be adaptive rather than being user-centered and carry out adaptation in accordance with the end-user preferences and context of use.

Hence, a responsive adaptation at runtime is still a challenge in the HCI field since there is no agreed technique for learning and executing the greatest adaptation rules in case of unanticipated situations during interaction. Thus, interfaces needs to be flexible and upgradeable over time considering contextual data accrued during interaction sessions, for instance the users satisfaction levels. Adapting interfaces emphasize their capability to fit new context supplies and to improve the users' experience for instance by reducing their frustration or improving their satisfaction levels.

The literature about HCI reports several implementations of adaptation, ranging from adaptability to adaptivity, and also including systems mixing both techniques [5, 23]. Adaptability concerns systems that allow the user to modify a number of parameters and adapt their behavior appropriately [19]. Systems that adapt automatically to the users preferences, in view of the system's assumption about their needs are called adaptive [23]. Both adaptation strategies have some drawbacks: adaptive interfaces can reduce the user workload but can outcomes a number of usability effects [12, 18]. Adaptable UIs maintains a high degree of user control; however, prior research has found that most of users are not willing to invest the efforts required for personalizing their UIs.

Recent advances in technological landscape as well as latest algorithms and real-time state assessments participate in changing the adaptation affinity. They open up the opportunities for more sophisticated adaptations that can recognize and overcome context requirement at runtime [4, 6]. The current adaptation challenge is to ensure a better understanding of context data to provide a meaningful guidance for the UI adaptation at runtime [17]. Adaption is required to respond to contextual changes efficiently and effectively ensuring a quick and agile reactivity. Adaptation is intended to shift for a proactive phase, which decides changes, anticipates difficulties and take steps to overcome them, while being guided by users preferences. We focus on the high-level scope of context-awareness and UI proactivity considering an agile paradigm to enhance the UI context-awareness at runtime.

Both HCI and Agile methods share the values of user-focus and iterative inspect-and-adapt cycles. However, there is still a

lack of interchange and integration between the different operated methods and disciplines. Similarities can be found in basic principles and practices as well as among the methods and tools that are typically applied [20]. In spite of this, there are still many challenges that must be overcome, which give us the opportunity to define a theoretical framework supporting adaptation at runtime with regards to agile principles aiming to:

(1) Establish the different units that relay to the whole adaptation practices and support diverse descriptions, implications and considerations of responsiveness.

(2) Provide an agile progressive enhancement of adaptation by ensuring the integration of different practices and techniques at runtime.

(3) Support the new urge of researches on intelligent UI to systematically outline (intelligent/proactive/agile) context-awareness based on the advantageous idea of decomposing adaptation for evaluation purposes revealed by Totterdell back in the 90's [29].

This paper is structured as follows: section 2 shows a literature review of related works. Section 3 identifies communalities between agile and HCI practices and their integration's challenges. Section 4 introduces the theoretical frameworks supporting agile context-awareness. Section 5 present and discuss results and future works.

II. RELATED WORKS

Agile approaches have gained a lot of attention in the software development field. It is defined as a methodology for the creative process that anticipates the need for flexibility and applies a level of pragmatism into the delivery of the finished product [11, 1]. In software engineering, agility refers to the viewpoint supporting mainly the capability for quick adjustment to changes in addition to the end-user involvement revealed at design time. We retain the definition of [25] considering Agility as the ability to act proactively in a dynamic, unpredictable and continuously changing environment. A survey on agile method [31] specified that agile methods are mostly used for Internet, back-end and front-end development project (figure 1). "These results suggest that while agile development is not confined to a particular type of software project, its inherent flexibility and responsiveness may be best suited for application that face rapid changes in both requirements and the facilitating technologies" [31].

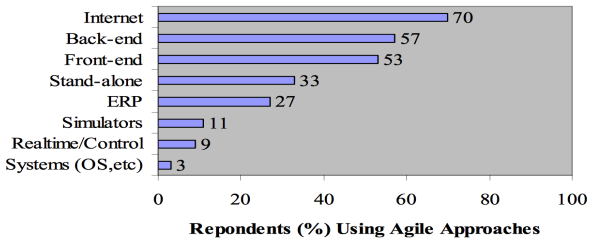

Figure 1. Project Types Supported by Agile Processes and Methods [31].

In this regard, agile methods were considered advantageous to support UI adaptation in an incremental, iterative and user-centered way [20]. Based on a common definition of agility, several HCI's works [20, 21, 24, 26] advanced agile method for UI development. Commonly investigation was aimed at bridging the gap between both disciplines HCI and SE. A significant overlap was identified, such as in iterative design, small releases and prototyping, scenarios, testing and evaluation. [24, 25, 26] demonstrate the contribution of agile paradigm for providing a beneficial support for HCI improvement within a user-centered paradigm. Commonly integration focused advancing UI development phase.

On the other hand, tailoring adaptation for users preferences is still the key factor for the improvement of UI usability [27]. Mostly, adaptations are performed when systems detect a context variation, by executing a particular reaction already encoded at design time. However, we argue that a successful Context-Aware Adaptation (CAA) [8] needs to be more proactive and more user-centered by meditating new accrued data during interaction in an incremental way. Two main concepts are required to be improved: first the user-centeredness, and second an incremental and iterative enhancement of adaptation. Both concepts match main agile practices.

Several analyses and studies targeted adaptive systems from different point of views, most of them focused on the dimensions of adaptation and were specific for distinctive domains [13, 15, 17] (medical, hypermedia, etc.). For instance, [14] proposed a classification for adaptive hypermedia methods and techniques by highlighting the adaptation process. Likewise [13] proposed a framework for categorizing UI adaptation based on two technical descriptions of two AUI key elements: the taxonomy of adaptation and taxonomy of triggers. Motti [9] proposed a generic framework for facilitating the development of context-aware application. The frameworks consist on two main parts, (1) the Context-Aware Design Space (CADS) that specifies analytical dimensions and their respective coverage levels for performing adaptation and [2] the Context-Aware Reference Framework (CARF) specifies dimensions and their possible instances for implementing adaptation. The CARF were intended to provide stakeholders an extensive list of possibilities to be considered while designing adaptations (figure 2). It represents a mind map composed by seven central branches aiding the implementation, execution and analysis of adaptation.

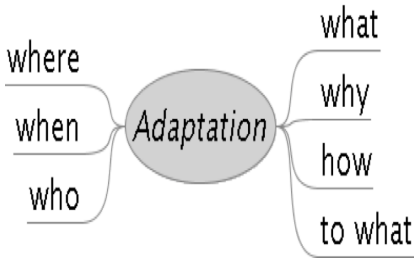

Figure 2. The reference framework CARF [9].

The most commonly cited issues with adaptive UI are the lack of flexibility, predictability, control, and privacy [7], mainly because those UI adaptations consider prior interaction knowledge (explicit context, domain models) [4, 6].

We are interested to extend the flexibility and provide system with the ability to learn and build novel knowledge in an incremental ways in view of context changes. We focused on investigating runtime context-aware adaptation in depth to identify key factor for developing and/or analyzing adaptation.

The idea behind adaptivity migrates from effortless flexibility into an intelligent responsiveness. Adaptations are expected to evolve continuously in a responsive and upgradeable way. Accordingly adaptation decisions should be determined throughout the system's lifecycle from early stages in term of guidelines and predefined adaptation, intended to inspire the system adaptation engine, until the execution phase when system is required to be scalable and flexible. Thus, in the current computational landscape, the support of intelligent runtime adaptation becomes a crucial requirement, which calls for a context-aware agile adaptation. These built-in agile practices and adaptation skills, should lead to a significant assimilation of the increased complexity. In this sense, we identified related challenges and proposed a theoretical framework supporting agile context-awareness.

III. AGILE PRACTICES MEETING CONTEXT-AWARENESS

A few years ago, the challenge of integrating agile methods with HCI was underestimated due to their differences in focus. However, nowadays there is a reasonable number of studies addressing this integration at design time, as can be seen in [20, 23, 25]. Both flexibility and agility are required to improve the UI adaptation. Each agility practice was widely discussed and showed a potential in different fields, for instance Software Engineering and User-Centered design.

We believe that the agile paradigm provides a beneficial support for HCI responsiveness at runtime as well as it was for design time. The goal is to reduce the gap between SE and HCI and consequently take advantages of agile practices to advance adaptation shortcoming at runtime. The main HCI requirement is still to improve the interaction and usability of the interface; which is valid and shared for different adaptation implementations. We examined agile practices and runtime/design time adaptations in terms of definitions, objectives and beliefs within their proper area, in order to underline fundamental concepts from diverse scopes. Based on such analysis we expand the reflection of intertwining agile practice with HCI. We focused on agile principles for UI adaptation by highlighting commonalities for both runtime and design time adaptation and then we overlapped their vision through an advanced UI context-awareness.

The context of use evolves over time; so adjusting UI to comply with new requirement proactively should be expected. Thereby 'Context-awareness" as well as 'user-centeredness' become crucial to improve the quality of interaction.

We acknowledge that tailoring relevant aspects and practices of agile paradigm and reproducing them for the UI context-awareness at runtime should show potential for improving adaptation proactivity. [26] Argues for the relevance of agile methods to improve systems usability defined as the extent to which a system can be used effectively and efficiently while satisfying a specified user.

An initial review of relevant works supporting agile practices for HCI development was conducted in [20, 21, 25, 26]. The significance of *human-centeredness* (HC) requirements to characterize agile methods [26], provided a starting point for reasonable assumptions about the effectiveness of agility in the HCI field. From an agile perspective, user requirements are particularly prone to change and evolution, as the software application evolves. This appears to address an important issue in HCI [26] and can provide great benefits at design time, however this requirement is continuing and exceeds the design level, which make it more worthwhile to enhance adaptation during execution.

Furthermore agile approaches often emphasize *iterations* as a requirement for the improvement of the software. As a result of improved iterativity and quick feedback, agile methods demonstrated its ability to support the successful software development. Similarly iterativity was recognized as the main design requirement to improve usability [26]. Such iterative development of user interfaces involves steady refinement of the UI *features* based on user testing, while the new trends of pervasiveness, and iterativity must be propagated and elaborated within the adaptation strategy in view of changing contexts of use and user preferences during interaction sessions.

Incremental paradigm would be another important aspect of agile approaches supporting a better knowledge transfer due to better user-system communication and frequent feedbacks from each iteration. Once again, the idea of incremental interfaces already exists, it consist on gradually increasing the UI complexity for a novice user by enabling advanced interface features incrementally as soon as the user needs and can use them. Such interfaces were developed for two intelligent learning environments: ITEM/IP [2] and ELM-PE [3]. Whereas this incremental aspect can be expanded to consider more context factors for instance the platform of interaction or the time. Further, incremental systems were based on predefined and static adaptation rules. Thus, they do not support the adjustment of UI complexity. To extend the consideration of the above-mentioned agility practices to a more practical perspective, all should be considered at runtime and established within the adaptation process. In this sense, agile methods are able to address major outlined UI responsiveness shortcomings, like considering individuals, their interaction preferences and changing context of use, besides emphasizing the importance of human factors for adaptation during interaction.

TABLE 1 SIMILARITIES BETWEEN AGILE PRINCIPALS AND HCI PRACTICES

AM basis	**Description**	**HCI Design practices**	**HCI runtime' practices**
Feature Driven	The system is segmented into sets of client-valued functionality, and development work is organized around producing these features.	Modelling tasks, Scenario	Modelling adaptation rules, Context models, Context tracking, Decision models, knowledge models
Iterative, incremental	Development is performed in repeated cycles (iterative) and in portions at a time (incremental)	Prototyping, user tests	Contexts evolution, Runtime adaptation, user tests, Prototyping UIs, Learning Knowledge
Customer involvement	The Customer Involvement gene means accepting changing requirements and including the user and/or customer feedbacks in the development	User test, User-centred design, user experience	User involvement, User centeredness, user implicit and explicit feedbacks, User commitments, personalization, controllable adaptability
Team Dynamics	The collection of "soft factors" and effects related to unique practices that influence the development team's performance	Design rooms, styles guides, collaborative design	Mixed-initiative adaptations, predictions, user controllable adaptability, System learnability
Continuous Integration	Continuous Integration involves methods of maintaining updated software.	Evaluation, Usability Inspections	Adaptability support, controllability, Iterative prototyping

A number of interdisciplinary interfaces during the different phases of SE and HCI developments could be considered, for instance: iterative modeling, evaluation, etc. The pervasiveness and the responsiveness of systems over heterogeneous contexts of use became the common significant requirement of both fields. Accordingly, HCI and Agile Software Engineering can converge into new shared principles and practices by using more methods known by both fields and by speaking the same languages [26, 20]. For HCI, existing evolution and improvement in artificial intelligence and machine learning fields provide more relevant and immediate adaptations, in spite of the challenges of new practical employments and applications, still resulting in the enhancement of the predictability and more user involvement.

Several matches were identified between SE and HCI fields and have promoted border crossing specially for development phases. Several researches were conducted to advance such cross-fertilization [11, 20, 22, 25, 26] and commonly they behave toward advocating the mutual benefits of exchange at development phase. Whereas HCI advances the UI improvement during runtime as well as development time. We argue that the benefit remain valid at runtime to improve UI proactivity. To better understand commonalities and bridge the gap between both fields, the Table 1 summarizes works on agility for adaptation development at design time [20, 25, 26] and contributes agility principles for runtime context-awareness. We outline similar practices identified in previous works [11, 20, 22, 25, 26] in the subsequent table, then we advance similarity between fields for UI adaptation at runtime. We consider main agile practices that were valuable for the development phase and we extend their support for runtime context-aware adaptation.

IV. THEOROITECAL FRAMEWORK FOR CONTEXT-AWARE AGILE ADAPTATION

This section shows the framework supporting agility for runtime context-awareness. The purpose of this work is improving the development of usable systems and the support of adaptation improvement, for instance learning interfaces, knowledge-based interfaces, and intent recognition. The main aim is to support stakeholders to develop and design systems that adapt at run-time and to make decisions concerning adaptation determinants, goals and rules. Further systems are intended to have the ability to accommodate up-to-date requirements, through a certain agile adaptation strategy.

Within this background, a particular importance should be accorded to the end-user involvement when determining and agreeing adaptation. The usefulness of human interventions consists mainly on allowing guidance, verification and improvement of the accuracy of adaptations. In this regard the framework contribute runtime UI adaptivity (similarly adaptability) by providing an abstract conceptualization that support adaptation while satisfying a user-centered paradigm.

In order to accomplish the above-mentioned requirements for a full understanding of different adaptation practices we proceed to characterize adaptation within two perspectives: an adaptation decomposition and agile arrangement.

Adaptation decomposition:

The first perspective consists of decomposing adaptation in a conceptual way of illustrating and simplifying the adaptation management process. The decomposition refers to the CARF [9]. This framework lists the most relevant concepts for implementing and executing context aware adaptation [9].

Six dimensions are maintained to identify adaptation features: (To what?, When?, How?, What?, Why?, Where?).

"To what?"

Recognizes the ontological context of use, the context is widely expressed as a triplet <user, platform, environment> and presents an abstraction of features involved during the execution influencing the interaction [9]. For instance the screen size as a feature provide guidance in the interface component numbers, widget sizes, colors etc.

The improvement of technologies summed with their availability, mobility and portability of devices support runtime pervasiveness and facilitate an advantageous UIs sensitivity for changing and heterogeneous context of use.

UIs should no more be designed for a regular situation (i.e. able-bodied user on a desktop computer in an office). However more cases need to be considered for adaptation with varied and heterogeneous situation. User can interact outdoor with a mobile device. In this case the environment have a high luminosity, the system should adapt the display parameters (contrast, screen luminosity) as well the interface colors and the size of components.

"*When?"*

Recognizes the involvement of systems in tracking the adaptation-triggers and support the decisions of adaptation. A trigger is a key element for adaptation and can be based several information classes that can be sensed, observed etc. It identifies mainly when to engage an adaptation. In the context of our works, the *"When?"* dimension refers to the taxonomy of triggers defined by [12], who propose a taxonomy of triggers classifying adaptation-triggers into five categories (figure 3): operator, system, environment, task and spatio-temporal. Such classification of triggers provides a support for the definition and the categorization of adaptations within the structure. For instance users can initiate adaptation. These cases represent an operator-based adaptation, where users can personalize or adjust the UI to their own preferences by means of feedbacks or a controllability feature.

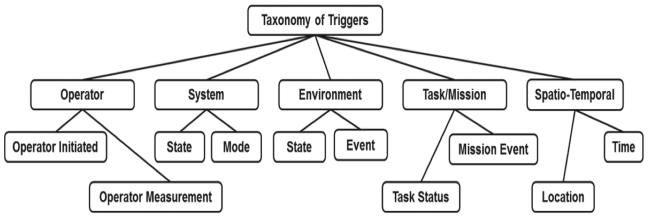

Figure 3. Taxonomy of Triggers for adaptive systems [12]

The triggering-process could be automatize regarding external element detected by means of sensors, for instance in case of a spatio-temporal triggers the location of user could provide guidance for the recommendation about restaurants, car rental agencies, hotels, stations. System based-triggers presents some internal facts initiating adaptation for instance: When the level of battery of a device is low the interface should adapt by changing appearance (i.e. colors, number of windows) or even functionality (i.e. connection, camera) to reduce consummation.

"How?"

This layer assigns certain adaptation constituents to specific adaptation determinants, for given adaptation goals.

An adaptation goal can be associated with a set of adaptation rules with different priorities satisfying a specific context of use. For instance, in the case of a user with dyslexia different adaptations could be executed; the system can decide adaptation by considering only the interface colors, or by changing the widget sizes and in case of a novice users the system can proceed at reducing the interface complexity.

This dimension regards adaptation methods and techniques at the conceptual and implementation level in term of guidelines, rules and learning algorithms.

"What?"

References the adaptation strategies (Rule's repository, Selection trees, Decision matrix) predefined and/or learned and acquired by systems during interactive sessions. In some case the representation of adaptation enable the users to intervene in the processing, usually by accepting, evaluating or rejecting the algorithm's decision. Figure 4 presents a decision tree developed in TRIDENT [30]. It presents a set of interactive tools that automatically generates a user interface for interactive applications.

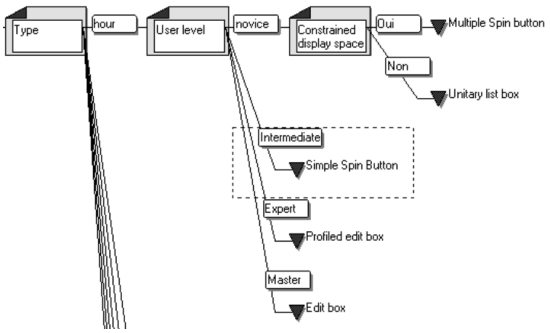

Figure 4. A partial view of the TRIDENT's AIO Selection Tree[29]

It includes an intelligent interaction objects selection based on three different concepts. First, a typology classifies abstract interaction objects to allow a presentation independent selection. Second, guidelines are translated into automatic rules to select abstract interaction objects from both an application data model and a dialog model. Third, these guidelines are encapsulated in a decision tree technique to make the reasoning obvious to the user.

"Why?"

This layer is intended to assist the information analysis. Several algorithms and scenario could aid adaptation decision and support reasoning, inferences and deal with complex or fuzzy information. Motti [8] provides a full overview about the current possibilities and scenarios for optimizing the context-aware adaptation of user interfaces with the application of machine learning algorithms. For instance Leiva [16] proposes the re-design of the UI components (widgets), based on the user interaction. Thus, the style of the widget is adapted according to the behavior of the user.

```
Algorithm: Leiva2012
Input: a set of widgets and their properties subjected to adaptation
Output: UI with adapted widgets
Begin
     Read_and_Parse_JSON_Widgets_Set();
     Select(widgets_set);
     Track_user_interaction(widgets_set, local_DB);
     For each (widget)
     {
         Update_score(widget);
     }
     Adapt(widgets,user_interaction);
End
```

The main benefits of applying and using intelligent techniques and machine learning consist in taking intelligent adaptation-decisions based on defined examples. Such examples can be used to find characteristics of interest (discovering), for instance by recognizing potential associations or patterns that are useful for predicting something. There are several ML algorithms that are capable of supporting context-awareness in its different phases and scenarios. Clustering can be used to identify relationships among context information, regression can be used to associate evaluation criteria, classification can be used to group contexts of use, and decision trees can support the selection of adaptation rules [8].

"Where?"

Recognize the Final UI which illustrates the adaptation effects regarding the "to what" requirements. The UI adapts in different manners through the modification of interaction for instance in graphical user interface adaptation concerns interface features focusing on the way of displaying information (e.g. colors, interactors, display size). In more innovative cases adaptation considers interaction styles and modality wich refers to the used sensory channel for information exchange (e.g. visual, haptic, auditory) and the interaction level defining the amount of interaction-control accorded to users regarding their experience.

The agile adaptation process

Figure 5 depicts the process of adaptation putting forward an example of HCI practices detailed in the above table. As well the figure highlight the iterative aspect and the arrangement of different identified adaptation features, outlining an agile outlook.

The design process is intended to be user-centered, iterative, and collaborative to improve usability and consistency. To achieve this, it is important to have stakeholders who can collaborate on UI design and user interaction issues from developers down to the end-users themselves. Agile UI adaptation implies that the adaptation is performed through different steps based on the above decomposition. The process adopts a user perspective, which challenges to make intuitive and consistent UI regarding user feedbacks. Within the Agile UI development, there is both a divergent and a convergent phase. In the divergent phase regarding the "When?" dimension, multiple adaptation triggers of "How?" adaptation strategies are invoked and presented for evaluation.

During the convergence phase, results from evaluations are used to create an ascertained adaptation rule ("What?") that is likely to be endorsed by user feedbacks. Gathered information are processed at the (Why?) dimension. The "Why?" layer concerns cognitions, learning, and intelligent features, which are responsible for the cognitive functionality of information processing.

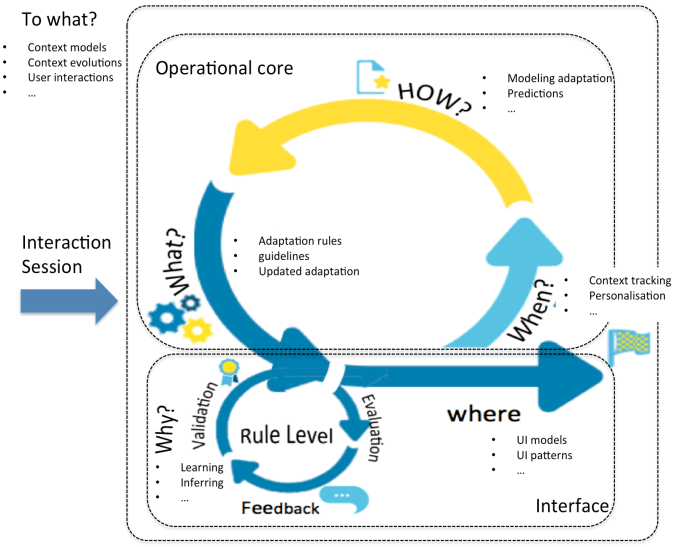

Figure 5. Agile runtime adaptation lifecycle.

For each step, the user behavior is tracked and then users are notified of what is going on and how the adaptation is progressing. They are able to add their input about their preferences in terms of implicit and/or explicit feedbacks. This can be advantageous for adaptation decisions to enhance the consideration of end-users choices. Such user involvement allows a closer distance between interface definition, adaptation decisions and end-users preferences.

A significant expected opportunity for the HCI and Agile domains to come together. They are both user-focused. They are both iterative and responsive to user feedback during iterations. Thus, combined with an agile process, user-centered design promises are enhanced by several advantages: (1) Regular adaptation to changing circumstances, (2) The simplicity and a better understanding of the problem, (3) A rapid testing and validation, provides a clear sociable and visual representation, besides improving usability. (4) An active 'user' involvement throughout the interface's development.

V. AGILE ADAPTATION IN THE SUPPORT OF METHODOLOGICAL PROTOTYPING

As we noted in previous section, we consider an agile method to support UI adaptation at runtime. This section aims to illustrate how the cross-fertilization of agile method and context-awareness can be of benefits for both HCI and SE communities.

We believe that the proposal could be used to support the Online Methodological Prototyping (OMP) with a high-fidelity level [29] for both fields. Commonly this task consists on defining the system as small releases evaluated by users and enhanced iteratively. Each iteration results new requirements that require additional cost (time and working development). This aspect remained a serious shortcoming of OMP.

Agile adaptation could contribute OMP and overcome costs shortcoming by advancing adaptation to upgrades systems. Enhancing adaptations and UI context-awareness initiate the need of deeply revising the current adaptation practices to account for (1) the alternative designs requisite for adaptation in the interface layer of system, (2) the parameters involved in driving adaptations (patterns, models etc.) and (3) the logic of adaptation at run-time defined by the operational core of the system.

Adaptations are managed by end-user during execution. Interaction session's results new knowledge that present next iterations requirements. Appropriate adaptations are evaluated and endorsed by end-users regarding their preferences and satisfaction.

An agile OMP with high-fidelity level supports vertical, horizontal and diagonal prototyping [29]. Figure 6 represent the OMP dimensions.

The horizontal prototype recognizes functionality that concern interface for instance by changing appearance and interaction style, colors, widgets and their arrangement ❶.

The vertical prototype targets more deep levels of system. For instance prototyping abstract models by allowing users to define and update explicitly their profiles in order to accommodate the appropriate adaptation regarding their preferences❷. As well vertical prototyping could concern operational core layers for instance by allowing end users to change the system complexity according to their expertise levels and evolution ❸.

The diagonal prototype combines both above stated strategies. In this case the system can learn to adapt. For example, by monitoring users interactive behaviors system can adapt deeper layers such as updating abstract models (interaction models, user profiles, design patterns) and/or upgrading the operational core by learning new adaptation rules, making adaptation decision and extending existing functionality.

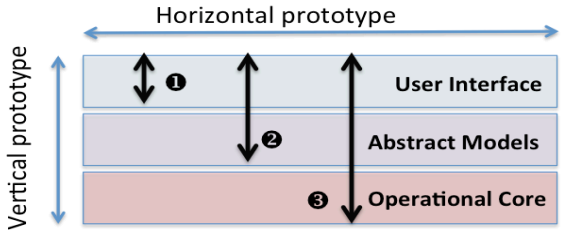

Figure 6. Agile Adaptation in the support of OMP.

As well agile adaptation could support the evolution of systems based on user experiences. For instance for an interface with vocal modality, the system needs to learn about users details during interaction (recognizing sounds, timber, volume, learning commands). In this case, agile adaptation support the training phase to enhance the system iteratively with regards to the users expectations.

VI. FINAL REMARKS

This article presents a theoretical framework integrating agile practices for UIs runtime context-awareness. We argue for extending the support of agile principles for HCI adaptation at runtime. The framework outlines a flexible lifecycle of agile adaptation considering six dimensions. Such decomposition allowed a unified and structured characterisation of adaptation regarding considerations, implications and strategies.

Each dimension was detailed and illustrated buy different concrete examples.

Our proposal is part of the Serenoa project addressing UI adaptation. Serenoa aims to provide a user interface (UI) that exhibits some capability to be aware of the context and to react to changes of this context in a continuous way.

The proposal extends CARF to support adaptation at runtime further than conceptualizing adaptation instances. The proposed framework supports the adaptation process from two perspectives: (1) conceptual instantiations referring to identified adaptation dimensions; (2) their arrangement within an agile topology outlining an iterative incremental cycle at execution.

On afterthought, an incremental enhancement capitalizes on the consideration of user feedbacks during interaction sessions that allow evaluation, promotion/demotion of rules and improvement of adaptation decisions. In the proposed framework the adaptation is user-centered, which implies that most of the times the user will be responsible for triggering or deciding upon an adaptation process.

Next, we will consider realize a methodological framework that considers conceptual, structural and procedural views. Further, we will take into account to define a methodological guidance that allows stakeholders to use the framework efficiently in several situations with examples integrating approach within concrete systems. Such guidance provides both HCI and SE communities a clearer vision about the benefits of agile context-awareness for different fields.

Likewise, we are interested in integrating our framework in typically agile methods used by SE communities. With the integration, examples will be required to explain the benefits of bringing the adaptation practices and runtime context-awareness on SE.

Finally, a platform prototype for the implementation of runtime context-aware adaptation is foreseen to validate the framework. With this prototype, we will be able to easier evaluate the interest and the usability of our proposal by conducing user experiments.

ACKONOWLEDGEMENTS

We are grateful to the Serenoa Project no. FP7- 258030, The Jounum project, the Louvain School Management and the Catholic University Of Louvain for their financial support.

REFERENCES

[1] Andersson, J, Baresi, L, Bencomo, N,. Software engineering processes for self-adaptive systems. In: software engineering for self-adaptive systems ii. springer berlin heidelberg, (2013). p. 51-75.

[2] Brusilovsky, P. Intelligent tutoring systems for World-Wide Web. In: R. Holzapfel (ed.) Proceedings of Third International WWW Conference, Darmstadt, Darmstadt, April 10-14, 1995, Fraunhofer Institute for Computer Graphics, (1995) pp. 42-45

[3] Brusilovsky, P., Schwarz, E., and Weber, G. ELM-ART: An intelligent tutoring system on World Wide Web.,,InternationalConference on Intelligent Tutoring Systems. (1996) 62-69.

[4] Chang, K., Hightower, J., and Kveton, B. Inferring Identity Using Accelerometers in Television Remote Controls. In Pervasive (2009), 151-167.

[5] Eisenstein. J, Puerta A, and R. Software. Adaption in automated user-interface design. Proc. of: The International Conference on Intelligent User Interfaces IUI (2000).

[6] Findlater, L., and McGrenere, J. Impact of screen size on performance, awareness, and user satisfaction with adaptive graphical user interfaces. Proc. SIGCHI Conference on Human Factors in Computing Systems CHI, ACM Press (2008), 1247-1256.

[7] Froehlich, J. et al. UbiGreen: investigating a mobile tool for tracking and supporting green transportation habits. In CHI (2009), 1043-1052.

[8] Genaro Motti, V,. Mezhoudi, N, Vanderdonckt, J (Machine Learning in the Support of Context-Aware Adaptation. Proceedings of the Workshop on Context-Aware Adaptation of Service Front-Ends (2012).

[9] Genaro Motti, V.. A computational framework for multi-dimensional context-aware adaptation. In Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems (2011, June) pp. 315-318.

[10] Glaiel, F, Moulton, A, et Madnick, S. Agile project dynamics: a system dynamics investigation of agile software development methods. (2013).

[11] Jameson, A. User modeling meets usability goals. In User Modeling Springer Berlin Heidelberg. (2005), pp. 1-3.

[12] Karen M. Feigh, Michael C. Dorneich, Caroline C. Hayes,. Toward a Characterization of Adaptive Systems Framework for Researchers and System Designer In. Human Factors: The Journal of the Human Factors and Ergonomics Society (2012) 1008-1024.

[13] Knutov, E., De Bra, P., Pechenizkiy, M. Generic Adaptation framework: a process-Oriented perspective. Journal of Digital Information, (2011).

[14] Knutov, E., De Bra, P., Pechenizkiy, M. AH—12 years later: a comprehensive survey of adaptive hypermedia methods and techniques. New Rev. Hyperme'd. Multime'd. 15(2009), 5–38.

[15] Lavie, T., & Meyer, J. (2010). Benefits and costs of adaptive user interfaces. International Journal of Human-Computer Studies, 68(8), 508-524.

[16] Leiva Luis. 2012. Interaction-based user interface redesign. In Proceedings of the 2012 ACM international conference on Intelligent User Interfaces (IUI '12). ACM, New York, NY, USA, 311-312. DOI=10.1145/2166966.2167028 http://doi.acm.org/10.1145/2166966.2167028

[17] Lim, B. Y., Dey, A. K. & Avrahami, D. Why and why not explanations improve the intelligibility of context-aware intelligent systems. CHI (2009), 2119-2128.

[18] Mackay we. Beyond iterative design: User innovation in co-adaptive systems. Rank Xerox, EuroPARC, (1991).

[19] Memmel,T., Gundelsweiler, F., Reiterer, H. Agile Human-Centered Software Engineering, the British Computer Society People and Computers XXI – HCI: Proceedings of HCI (2007).

[20] Noor, R. Rabiser, P. Grunbacher, Agile product line planning: A collaborative approach and a case study". Journal of Systems and Software vol. 81, (2008) pp. 868-882.

[21] Obendorf, H and Finck, H, "Scenario-Based Usability Engineering Techniques in Agile Development Processes," Proceedings CHI EA'08 CHI'08, New York, (April 2008), pp. 2159-2166.

[22] Oppermann, R., & Rasher, R. Adaptability and adaptivity in learning systems. Knowledge transfer, 2(1997), 173-179.

[23] Owen, R., Koskela, L.J., Henrich, G., & Codinhoto, R. Is Agile Project Management Applicable To Construction?. In: Sacks, R., and Bertelsen, S. (ed.), Proceedings 14th Annual Conference of the International Group for Lean Construction, Ponteficia Universidad Catolica de Chile, Santiago, Chile, (2006). 51-66.

[24] Patton, J. "Designing Requirements: Incorporating User? agile-Centered Design into an Agile SW Development Process," Extreme Programming and Agile Methods— XP/Agile Universe, Vol. 2418, No. (2002)pp. 95-102.

[25] Silva da Silva, T, Selbach Silveira, M, Maurer, F, Hellman, T, User Experience Design and Agile Development: From Theory to Practice, Journal of Software Engineering and Applications, (2012), 5, 743-751.

[26] Sohaib, O., & Khan, K. (2010, June). Integrating usability engineering and agile software development: A literature review. In Computer design and applications (ICCDA), 2010 international conference on (Vol. 2, pp. V2-32). IEEE.

[27] Terada, T., Masahiko, T., Keisuke, H., Tomoki Y, Yasue , K., Atsusshi K, and Shojiro, N. G., Ubiquitous Chip: A Rule-Based I/O Control Device for Ubiquitous Computing. Pervasive (2004), 238- 253.

[28] Totterdell, P., Boyle, E.,. The evaluation of adaptive systems. In Browne D., Totterdell P., Norman M. (Eds.): Adaptive User Interfaces. Academic Press;(1990) p. 161-194.

[29] Vanderdonckt, Jean, Coyette, Adrien, et al. Modèle, méthodes et outils de support au proto-typage multi-fidélité des interfaces graphiques. Revue d'Interaction Homme-Machine Vol, 2007, vol. 8, no 1.

[30] Vanderdonckt, J. M. and Bodart, F. Encapsulating Knowledge for Intelligent Automatic Interaction Objects Selection, in Proc. of InterCHI'93. 1993: ACM Press.

[31] Vijayasarathy, L. E. O. R., & Turk, D. (2008). Agile Software Development: A survey of early adopters. Journal of Information Technology Management, 19(2), 1-8.