International Journal of User-System Interaction 14(1) 2021, 1-29

GISATIE: A User Interface Adaptation Life-Cycle

Víctor López-Jaquero[1], Vivian Genaro Motti[2], Francisco Montero[1], and Pascual González López[1]

[1]Laboratory of User Interaction and Software Engineering (LoUISE), Computer Science Dept., University of Castilla-La Mancha, Campus Universitario, 02071 Albacete, Spain

[2]Dept. of Information Sciences and Technology (IST), Volgenau School of Engineering, George Mason Univ., Engineering Building, Rm. 5350, Fairfax Campus, USA
E-mail: VictorManuel.Lopez@uclm.es, fmontero@dsi.uclm.es, vmotti@gmu.edu, Pascual.Gonzalez@uclm.es

Abstract. Adapting the User Interface of an interactive application consists in modifying its different elements according to various levels of granularity. Adaptation aims at addressing specific needs, wishes, and requirements either of a particular user or a group of users. While user interface adaptation has been extensively studied, in particular for context awareness, one of the most widely used adaptation life cycles is Dieterich's survey of adaptation techniques. This survey considers only the execution part of the adaptation lifecycle and involves only one actor, user or system, in each adaptation stage. To overcome these shortcomings, we introduce GISATIE, a user interface adaptation life-cycle specifying which agents are involved in each adaptation stage: goals, initiative, specification, application, transition, interpretation, and evaluation.

Keywords: user interface adaptation, adaptation life-cycle, adaptation process, profile

DOI: 10.37789/ijusi.2021.14.1.1

1. Introduction

Adapting the User Interface (UI) consists in modifying its different components according to various levels of granularity, ranging from low level (e.g., changing the color of a highlighted UI element) to high level (e.g., reformatting the layout of a dialog box). Adaptation aims at addressing specific needs, as well as wishes and requirements either of a particular user or a group of users. Adaptation can be classified according to two categories depending on who controls it (Benyon and Murray, 1993): *adaptability* occurs when the end user adapts the UI, while *adaptivity* occurs when the system has the capability to adapt the UI. *Mixed-initiative* adaptation (Horvitz, 1990) exists when both, the end user and the system, cooperate

towards achieving the UI adaptation. Adaptivity, although more expensive to develop, demonstrates some benefits (Lavie and Meyer, 2010) and it is used in a wide range of domains of human activity, such as ambient intelligence (Escribano et al., 2008), automotive systems (Rogers et al., 2000), electronic commerce (Sherman et al., 2003), algorithmics (Kerren and Stasko, 2002), and information systems (Dieterich et al., 1993).

The main shortcomings of adaptivity are (Lavie and Meyer, 2010; Bunt et al., 2004): the end user disruption caused by an unexpected behavior for the end user and the cognitive perturbation when the end user, confronted with a new UI, must reconcile with this UI by imagining the correspondence between the UI before and after adaptation. Between these two UIs, there is usually nothing but a big gap, which reinforces the cognitive perturbation. Cognitive psychology (Gardiner and Christie, 1987) refers to this phenomenon as a cognitive "destabilization", meaning that any user is de-stabilized when confronted with anything unexpected, unprecedented, or unpredicted contents. The end user remains in this stage of cognitive destabilization until a "restabilization" restores a relation between the past and the newly presented contents. The end user does not suffer from these shortcomings in adaptability since the user remains in control and therefore knows what is subject to adaptation, as opposed to adaptivity where the system is in control and the end user may not know what the system is doing and why. To address this challenge, animated transitions (Dessart et al., 2011; Huhtala et al., 2009; Huhtala et al., 2010; Schlienger, 2007) can show how the adaptivity has been conducted, what has been adapted, and even why. In this sense, using mixed-initiative during the adaptation can help also in making better decisions by letting the user decide in conflicting situations where the adaptation engine is not sure about how to proceed.

The current progressive migration of interactive applications from desktop computers to mobile devices changes the interaction habits of the users. A new cohort of neophytes is becoming more attracted by the abilities of interactive applications to support many daily tasks, such as buying flight or theater tickets. Simultaneously, the complexity level of the applications and the amount of information available are quickly increasing. To this end, adaptation techniques that adjust the application features according to the context information should be devised (Schlee & Vanderdonckt, 2004), for which many adaptation techniques are applicable (Firmenich et al., 2011; Calvary et al., 2003; Langley, 1999; Norman, 1986).

Nowadays, one the most widely used understanding of the adaptation life-cycle comes from Dieterich's survey on adaptation techniques (Dieterich et al., 1993). Despite being produced in 1993, this taxonomy suffers from several shortcomings: it is constrained to a single entity (e.g., the user and the system) in each stage of the adaptation life-cycle, it does not handle explicit collaboration during the different adaptation stages and it is restricted only to the execution part of the adaptation. Furthermore, some of the most relevant issues in the adaptation, such as how the adaptation is specified, were left out of the framework (Motti and Vanderdonckt, 2013). In particular, Dieterich's taxonomy is incomplete with respect to the seven stages of Norman's Theory of Action (Norman, 1986). This theory describes how a user interacts with an interactive application from the very beginning, from forming an intention to reach a goal, until the end, when evaluating the results from the actions taken to achieve that goal.

This paper revisits Dieterich's taxonomy by defining stages based on the mental model proposed by Norman (1986). These stages aim at improving the user involvement in the adaptation life-cycle and foster a detailed description of how the adaptation life-cycle is carried out. By doing so, we aim at covering the whole adaptation life-cycle, and not just the execution part of adaptation. This life-cycle is complemented with an adaptation profile that serves as a quick reference to document the coverage and actors involved at each stage of the adaptation life-cycle. To illustrate the stages of our life-cycle, a multi-agent system will be used as a running example, together with some other examples at some specific stages of the adaptation life-cycle.

This paper discusses some related work for user interface adaptation in Section 2, with a special focus on frameworks and taxonomies for adaptation. This related work section includes also some multi-agent systems closely related to our own one, which is used as a running example throughout the paper. Then, GISATIE, the adaptation life-cycle is presented in Section 3, and illustrated with some examples. Some conclusions and future work are reported in Section 4.

2. Related Work

This section discusses those adaptation classifications, taxonomies and frameworks that were used as the foundation of our contribution, together with some relevant multi-agent system aimed at providing adaptation support.

2.1. Adaptation Classifications, Taxonomies, and Frameworks

Characterizing adaptation is not an easy task, and unfortunately in many systems, there is not such a clear underlying framework for the design of the adaptation capabilities. Throughout the years several approaches for describing and classifying adaptation have been proposed. This section is not intended to make an extensive review of those techniques supporting UI adaptation, but instead it compares the expressivity of any classification scheme that could serve for expressing UI adaptation life-cycle, such as classifications, taxonomies, and frameworks. A review of those adaptation classifications, taxonomies, and frameworks were used as the foundation of our contribution is presented in a chronological order.

Browne's classification for user adaptation (1986)

Pioneering work started with a Command Language Grammar (CLG) for specifying an adaptive UI (Browne et al., 1986). They concluded that the major strength of CLG for this purpose was its support to the Principle of Separation of Concerns, starting with a conceptual model of the adaptive UIs and enforcing this model throughout the rest of the development life cycle. Although this enforcing was made explicit, it was not obvious how to easily propagate all the specifications aspects contained in the CLG specifications into the final code. In particular, they indi-cated that CLG has very limited facilities for expressing the presentation and behavior of a UI, thus raising the need for improving the CLG expressivity in this respect.

Cockton's Parametrization strategy for Adaptation (1987)

Cockton (1987) presents an argumentation regarding how to achieve, in the UI, the required flexibility to provide adaptation facilities. This flexibility is provided in terms of parametrization. All UI constructs should be as parametrized as possible, so the behavior of each construct can be personalized as much as possible too. This parametrization is not just applicable to the UI elements, but also to those control classes that rule the functional behavior. Cockton also provides some ideas regarding how actual adaptation can be carried out. He proposes three options. The first one is enabling, which consists in having in a single dialog several interaction paths available, and therefore the different paths can be chosen by enabling and disabling the different options that lead to a path, according to changes in the

context of use. The second one is switching, which provides several different dialogs for a single task, and the adaptation engine switches from one version to another according to the changes in the context of use. Finally, the third option is reconfiguring, which allows reconfiguration of several parameters of the UI interaction. Thus, the UI is adapted by adjusting the values for these parameters. Regarding the life-cycle, a two-stage approach includes: *diagnosis*, in which the system guesses the needs and skills of the user based on recorded data and a *treatment* to find a remedy for the detected situation.

Norcio and Stanley Survey (1989)

This survey defines adaptation as a process based mainly on the knowledge of (Norcio & Stanley, 1989): the user, the interaction scheme, the task and the system. Although models can be used to specify this knowledge, this is considered a complex task. For instance, the users have different profiles, wishes, and requirements, they comprehend and process information in different ways. Although the main challenge in 1989 was still the lack of technology to support the UI adaptation, some issues highlighted remain still open nowadays, such as: systems are developed for the average user (usually able-bodied in a stable environment with a desktop), which historically has been forcing users to adapt themselves to the system, while the opposite should occur; the increasing availability of computers and consequently of novice users; the potential loss of control by the users caused by inconsistencies and incoherencies in an adaptive application and the costs and complexity naturally added to implement and provide adaptation.

Totterdell et al.'s classification for user adaptation (1990)

Adaptation techniques (Totterdell et al., 1990) are inspired by the different variants in adaptation explored by using Prisoner's Dilemma. The classification is made in terms of the level of adaptivity they present, that is, the amount of control that a system has in negotiating a change: designed systems, adaptable/tailorable, adaptive, self-regulating, self-mediating and self-modifying. This work interprets UI design as an adaptation problem to construct artifacts that are well adapted to their environment. These levels also reflect how some design decisions, usually made by the designer, are progressively transferred to the system as we progress in the level of the taxonomy. Three actors are considered: the designer, the user and the system.

Dieterich's taxonomy for user adaptation (1993)

Dieterich's taxonomy (Dieterich et al., 1993) of UI adaptations has always been considered as a seminal reference for classifying different types of UI adaptation configurations and techniques. This work sorted more than 200 papers dealing with various forms of UI adaptation and summarized them into four stages needed to perform any form of adaptation. The *initiative* stage involves the user or the system to suggest its intention to perform an adaptation. The *proposal* stage states that, if a need for adaptation arises, proposals of adaptation should be made that could be applied successfully given the current situation. In the *decision* stage, as several proposals could emerge from the previous stage, which adaptation proposal best fits the needs for adaptation should be decided. The *execution* stage will actually execute the adaptation chosen at decision stage. The authors classify every system with adaptation capabilities according to the actors involved at each stage.

Oppermann's survey (1994)

The control given to the user regarding the timing and the contents of the adaptation should be emphasized (Oppermann, 1994). Adaptation can be achieved in a shared initiative, shared decision-making, or shared execution between the user and the system, i.e., by combining adaptivity and adaptability, a combination is more promising than using adaptivity and adaptability alone. Adaptation should convey its rationale to the end user (e.g., with a tutorial), the selection and definition of adaptation opportunities, an overview of performed adaptations, and the possibility of subsequent changes in the concluded adaptations. Adaptation comprises three stages: *afferential* (the gathering of the context information, user interaction), *inferential* (the processing and inference of this information), and *eferential* (the implementation and presentation of the adaptation to the end user).

Brusilovsky (1996)

Brusilovsky (1996) conducted some work on adaptive hypermedia, which is one particular form of adaptation, but not the only one. This work focuses on adaptive hypermedia, and also does not cover the web techniques entirely. The work reports an extensive review of adaptivity techniques for web applications. While this extensive effort classifyies and structures techniques, it is not explicitly based on a notion of the context of use, defined as user,

platform, and environment (Calvary et al., 2003). The adaptation is analyzed concerning "adaptation of what with respect to what", but the context of use is not fully exploited in this way. Furthermore, other dimensions of adaptation (e.g., when, how, with which constraints) are not extensively researched, thus raising the need for a multi-dimensional framework for adaptation.

Lorenz's Methodology of Adaptive Systems (2000)

The Methodology of Adaptive Systems (Lorenz et al., 2000) introduces a point of view about adaptation quite different from Dieterich, although it is aimed at context-aware systems only. Three stages are identified similarly to Lorenz (2000). The first one is *afference* where the user behavior is observed. Then, the system records how the user acts and reacts and gather information regarding user data, such as physiological characteristics, references, interest, personality knowledge and expertise or user similarities and differences. The next stage is *inference*, where the data gathered by the previous stage is analyzed according to model assumptions on user needs, heuristics or ontology models (Furtado et al., 2001) of the application domain. Finally, the *efference* stage executes the adaptation activity decided in the inference stage.

Brusilovsky's taxonomy (2001) and extension (2006)

Brusilovsky (2001) presented a new taxonomy that classifies adaptive hypermedia systems, considering adaptation of two major aspects: *presentation* (e.g., multimedia presentation, text presentation and modality) and *navigation* (e.g., direct guidance, link sorting, link hiding, link annotation, link generation and map adaptation). This taxonomy supports classifying some adaptive systems. Thus, Hanisch et al. (2006) extended this taxonomy to accommodate also multimedia components. As web technologies evolve, not only text fragments or links can be adapted, but also movie clips, 3D graphics, and so on. The adaptation methods and techniques were extended and also classified by component types: models, views, controllers, widgets, graphic items, movie clips, scripts and strategies.

W3C Authoring techniques for device independence (2004)

This W3C recommendation gathered a series of adaptation techniques, but mainly for adaptation with respect to the computing platform (device and web browser). While this dimension represents an important constraint in conducting adaptation, it is certainly not the only one. The structure of the

document composed by types of techniques could be expanded into more refined categories and sub-categories, with more links to existing techniques.

McKinley's Taxonomy for Compositional Adaptation (2004)

McKinley (2004) proposes a taxonomy for compositional adaptation of software. Compositional adaptation results in the exchange of algorithmic or structural parts of the system with ones that improve a program's fit to its current environment. This kind of adaptation is based on the separation of concerns between the functional behavior and the cross-cutting concerns, the computational reflection that provides a vehicle to query the different aspects of a system, the component-based design practices that enable the development of the different parts of a system separately and the middleware that usually provides the compositional capabilities.

McKinley's taxonomy uses three dimensions to describe those systems supporting compositional adaptation: 1) how to compose, 2) when to compose and 3) where to compose. The first dimension describes how composition is implemented. This dimension can be carried out by different entities (composers): a human (software developer, administrator), a component loader, a run-time system or a meta-object. When to compose dimension describes when the adaptive behavior is composed with the functionality. It can be either *static* or *dynamic*. If this dimension is carried out at development time, compile or link time, or load-time, it is said to be *static*. On the other hand, if it is made at run-time, *dynamic* composition appears. Where to compose dimension describes where in the system the adaptation code is inserted. The most common approach is to place the code in the middleware, although extensible systems have also been used. Although this taxonomy is applicable to any software as a whole, and not just its UI, we believe that the characterization of the various dimensions (i.e., how, when, where) is particularly constructive for UI adaptation.

Arhippainen's Design Space for adaptation (2009)

Arhippainen (2009) defines a design space for adaptation according to the triple: target, means and time. The target refers to with regard to what the adaptation is performed, e.g., to the users, to the environment, or to the platform aspects. The means refers to the software components involved, such as: task models or rendering techniques. And the time defines a static or

dynamic adaptation (between sessions or occurring during run-time). Although this definition is precise, it is too limited, once it does not include important aspects like what is being adapted, or how.

Gomez's survey of adaptation techniques (2009)

Many different research communities have been proposing approaches to implement adaptation (Gomez et al., 2009), and Hypermedia System and Intelligent Tutoring Systems in particular. The task of adaptation breaks down to a mediation between resource provision and re-source demand. In doing so, it is necessary to obtain some representation of them, either directly or through intermediate models that can be further processed to achieve this information. Correspondingly, major differences in adaptation approaches manifest themselves in the employed sources, the way they are represented and the techniques used to derive the user demand from them. Therefore, that survey was structured according to these model-related aspects.

Paramythis et al.'s framework (2010)

The authors proposed a framework serving as a guide for layered evaluation of adaptive interactive systems (Paramythis et al., 2010). This approach decomposes the system into layers, i.e., collect input data, interpret the collected data, model the current state of the "world", decide upon adaptation and apply adaptation, that can be evaluated independently using a set of formative methods. The authors then addressed the when, why, how questions for web sites and hypermedia systems, but not for any type of UI.

Abrahão et al.'s reference framework (2021)

A conceptual reference framework is defined for intelligent user interface adaptation containing a set of conceptual adaptation properties that are useful for model-based UI adaptation. The objective of this set of properties is to understand anymethod, to compare various methods and to generate new ideas for adaptation. This framework is decomposed into four parts: the context of use, the software system, the intelligent UI adaptor, and the external sources. This framework mainly serves as a reference for structuring components in the software architecture of an intelligent system equipped with an intelligent UI adaptation. It discusses a set of questions for any UI: who, what, why, how, to what, when, where (Motti & Vanderdonckt, 2013).

2.2 Discussion

Current taxonomies and frameworks for adaptation are incomplete and suffer from several shortcomings. Cockton (1987) provided some ideas regarding two main aspects of adaptation: how to perform the actual adaptation to modify the UI and how to provide the required flexibility by parametrization of UI constructs. However, the process for adaptation proposed by Cockton covers just the detection of the needs for adaptation and the execution, but very relevant stages, such as the selection of the appropriate adaptation for a given situation were left out of the process. Norcio et al. (1989) introduced some relevant issues found in adaptation, such as the fact that most applications are designed for the aver-age user, forcing the user to adapt to the application. The work of Browne et al. (1986) provided very interesting insights regarding the use of an underlying formal foundation for adaptation. Nevertheless, CLG lacks expressivity for the UI behavior and presentation.

Dieterich's taxonomy only considers two entities (i.e., the user and the system) in each stage of the adaptation life-cycle, it does not handle explicit collaboration and it is restricted to the execution part of the adaptation only. Furthermore, some of the most relevant issues in the adaptation life-cycle such as how the adaptation is specified were left out of the framework. In Totterdel's work (1990) one extra actor responsible for adaptation (i.e., the designer, the system and the user) is considered, but no explanation on how they can collaborate to carry out an adaptation life-cycle stage is provided. They proposed also a taxonomy for adaptive systems. Nevertheless, it relies only on the amount of control the system has on adaptation. On the other hand, the methodology for adaptive systems proposed by Lorenz et al. (2000) introduc-es a framework mostly aimed at context-aware systems, and it also suffers from those short-comings identified for Dieterich's framework. Again, relevant issues in adaptation, such as how the adaptations should be specified or how do they comply with the original intend of the adaptation life-cycle are left out of the framework. Similarly to Lorenz et al. (2000), Oppermann's (1994) adaptation life-cycle only covers some stages of the adaptation life-cycle. However, very important stages, such as the evaluation of the adaptation or how the adaptations are specified are not considered.

Compositional adaptation, as proposed by McKinley et al. (2004), provides some interesting insights in the characterization parameters that any adaptation framework should consider. More concretely, it illustrates how the

actual adaptation should be applied, describing how the changes required in the application for the adaptation should be included, at what time (compile-time, load-time, development time, etc.), and finally it points out where in the original application code the adapted code should be placed. Unfortunately, this work is aimed at general software adaptation rather than UI adaptation. Time component of adaptation is considered in Arhippainen's (2009) work.

Regarding what can be adapted, the taxonomy proposed by Brusilovsky (2001) and its extension (Hanisch et al., 2006) provide a foundation for the adaptation designer to know what can be actually adapted. In this sense, the W3C Authoring Techniques for Device Independence contribute also to help in answering the what question in adaptation.

Furthermore, all these approaches are incomplete with respect to the seven stages of Norman's theory of action (Norman, 1986). This theory describes how a user interacts with an application from the very beginning, when the user is forming his intention to reach a goal, until the end, when the results from the actions taken to achieve the goal are evaluated. The adapta-tion process should cover all seven stages of Norman's theory of action, covering aspects of adaptation so important such as evaluation or the transition from the original system to the adapted one (López-Jaquero et al., 2007).

Otherwise, an incomplete adaptation life-cycle will happen, thus ignoring very relevant issues in adaptation. Finally, the actors involved at each stage must be properly described to characterize any system with user adaptation facilities. How these actors collaborate to achieve an adaptation stage should be also considered to describe the adaptation stages in a life-cycle.

2.3 Related Multi-Agent Systems in Adaptation

Next, four relevant multi-agent systems aimed at providing adaptation capabilities are dis-cussed. Multi-agent systems have been already used to provide adaptation capabilities. The next section reviews four of them. This review is focused on multi-agent systems designed for adaptation.

MASHA (2006)

MASHA (Rosaci et al., 2006) introduces a MAS aimed at supporting adaptivity in web sites. MASHA aims mainly at supporting recommendation when browsing the web. Nevertheless, it also supports adaptation to the device the user is currently using. Adaptivity is based on the information collected by a client agent running in the different user's devices. The

interests that a user have in a concept help in the recommendation task as well. Moreover, in the rec-ommendation task, collaborative filtering is employed; therefore the profiles of other users are also used during this recommendation task. There are three agent types: 1) MASHA client, which collects the information about the user; 2) MASHA server, which builds the profile out of the information provided by MASHA client type, and lastly, 3) MASHA adapter, which adapts a website according to the user profile and the device he is currently using. This agent delivers to the user concept instances whose representations are compatible with the device size.

Benaboud and Sahnoun (2006)

Benaboud and Sahnoun (2006) proposed another multi-agent system that uses three models to drive the adaptation, namely a system model, a user model and a presentation model. The adaptation life-cycle considered in this approach includes three stages: 1) user characteristics inference, 2) filtering and selection of the appropriate information resources and 3) organization of the presentation of the resources in the available space. This approach presents some limitations regarding our definition of an adaptation life-cycle. First, the adaptation life-cycle is focused on the execution part of adaptation, all the evaluation part is left out. Moreover, only the user is considered in the context of use. This approach is not aimed at supporting generic adaptation, but it is aimed at adapting just the way in which the resources are presented.

ADUS (2007)

Mitrovic et al. (2007) use also a MAS for adaptation, namely ADUS. The main difference here is the use of mobile agents. Mobile agents travel to client platforms on behalf of the ad-aptation framework to maximize the portability of the MAS. The adaptation stages are: 1) adapting an abstract UI definition to a concrete platform, and 2) monitoring user interaction and communicating this information to other agents. There are three main agent types in this approach: 1) visitor agent, which is a mobile agent than brings a service to the device and generates an appropriate abstract specification for that service, 2) user agent, which collects information from the user and the device he is using and acts as a proxy for visitor agent and, 3) ADUS agent, which is the actual entity performing the adaptation of the UI according to the information gathered by a user agent. XUL language is used for the abstract UI.

AB-HCI (2009)

AB-HCI (López-Jaquero et al., 2009) is a MAS aimed at supporting an adaptation life-cycle based on (Dieterich et al., 1993). To do so, a representation of the running UI is described in terms of a UI description language, namely UsiXML (Limbourg et al., 2004). The system is designed so as to support the execution of generic adaptation rules, to improve the flexibility of the system. The system receives the information from the context of use by means of software and hardware sensors in the client device. The information gathered by means of sensors is sent to AB-HCI regularly by a back-ground application running on the client device. The MAS then selects the best adaptation to apply according to the changes in the context of use reported.

Discussion

Although MASHA provides support for the adaptivity of web sites, it presents some limitations. First, the adaptation life-cycle does not consider the evaluation part. There is no support for generic adaptation rules, instead the adaptations are hardcoded. Thus the flexibility of the system is compromised. In ADUS some limitations can be identified. First, it focuses on the execution of the adaptation, and the evaluation of its results is not considered. Second, the language used for the specification of the abstract UI supports only the specification of the abstract UI in terms of widgets and layouts. Thus, given that ADUS does not represent other important UI facets, such as the domain model and their respective task per widget, it constraints the possible adaptations to widgets and layouts, excluding their semantics. Lastly, it does not support the specification of generic adaptation rules. Therefore, the reusability of the adaptation framework is also compromised. Although AB-HCI is already a flexible and powerful approach, it supports only partially the stages of the gulf of execution, and it does not support the gulf of evaluation at all (see Figure 1). Once the limitations of some of the most closely related MAS applied to adaptation were identified and presented, our MAS, named MAYA (Multi-Agent sYstem for Adaptation) will be used as the main running example with additional examples for some stages.

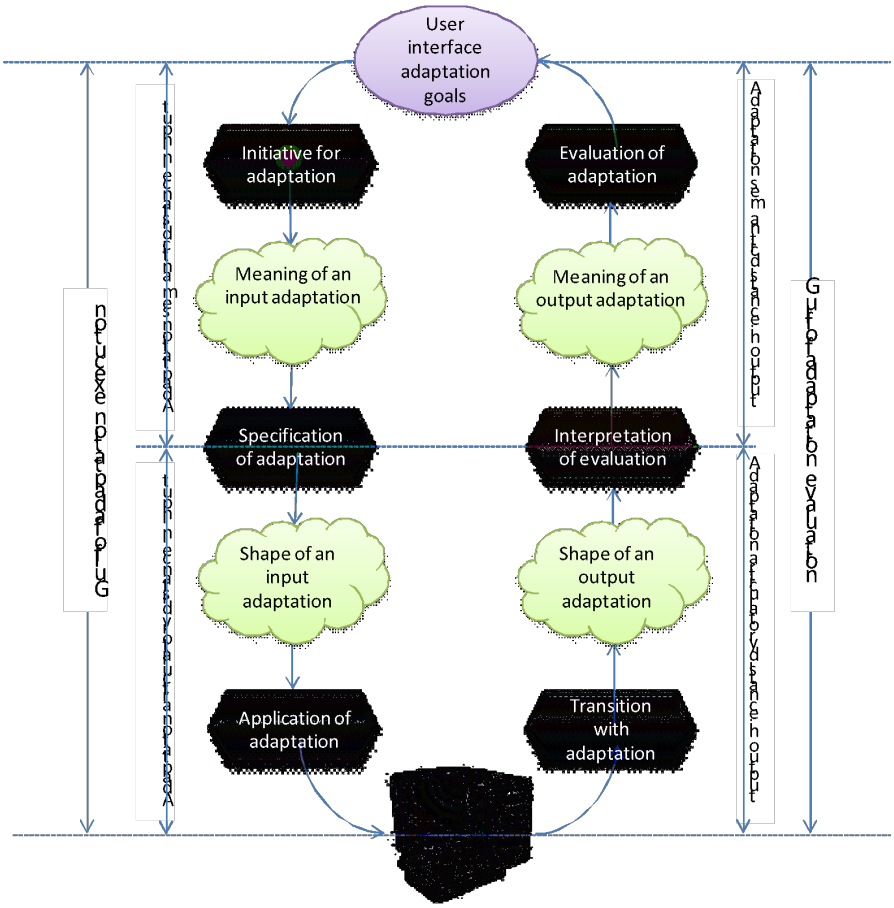

Figure 1. GISATIE framework for adaptation stages

3. GISATIE: A User Interface Adaptation Life-Cycle

To overcome some of the current limitations identified in the adaptation models and taxonomies previously analyzed, an adaptation framework has been devised. This framework aims at covering all the stages identified in adaptation, and not just those related with the execution of the adaptation, as in the approaches previously discussed. This framework is an extension of

ISATINE (López-Jaquero et al., 2007) by refining the stages and adding the agents responsible for conducting each stage.

GISATIE (*Goals, Initiative, Specification, Application, Transition, Interpretation and Evaluation*) adaptation framework comprises seven stages, as a result of the specialization of the seven stages found in Norman's mental model of action. These seven stages are illustrated in **Error! Reference source not found.**. Norman's theory of action (1986) and our GISATIE adaptation framework are both about what end users do within a cycle of interaction with a UI. Our adaptation framework is also about design, about how adaptation steps support end users in performing sensory, cognitive, and physical actions in order to carry out their interactive tasks.

The seven stages of GISATIE can also be classified according to the part of the adaptation life-cycle where they occur. In this sense, two parts are identified, namely the gulf of adaptation execution and the gulf of adaptation evaluation (see **Error! Reference source not found.**). The first gulf is related to all the stages in the adaptation life-cycle required to finally execute the adaptation. On the other hand, the second one encompasses those stages that lead to finally be able to assess the adaptation that results from the adaptation execution gulf.

All seven stages in GISATIE adaptation life-cycle can be carried out by either one entity or several ones in collaboration. Next, what entities (or actors) are considered in the framework and how these entities can interact is described. Three different entities are considered at each stage of GISATIE: the *user* (U), a *machine* or a system (M) and a *third-party* (T). The entity *user* represents any user interacting with the application where the adaptation takes places. The *machine* entity represents any hardware platform involved in the interaction, including PC, mobile devices such as a tablet or a smartphone or any kind of robot. We are using the broader term Machine rather than System, since in our framework any autonomous machine can play a role in the adaptation life-cycle. The entity *third-party* corresponds to those stakeholders that can be involved in the interaction, but being external to the couple machine-user. Notice, that in collaborative systems, there can be several users interacting with the several machines, or even several users interacting with a single machine. Each stage can be carried out by either the user, the machine or a third-party. Nevertheless, it is also possible that any of the stages are performed coordinately. Five different coordination modalities are allowed:

1. *Negotiation*: in this coordination modality, options are presented by each entity and the final result is negotiated between the entities so as to reach a consensus. T could serve for this purpose when, for instance, contradictory output is produced by U and M, or for stating which entity has priority.
2. *Consultancy*: when an entity estimates that it does not have information or responsibility enough to achieve the adaptation stage, it may request help/support from any other entity to achieve its purpose. When the results come back to the requesting entity, it decides the final option, thus keeping the control over the decision.
3. *Delegation*: this modality is the same as consultancy, but without any return to the requester. The requested entity takes the decision and may send a notification.
4. *Coopetition*: this is a form of collaboration where at least two entities should compete while cooperating at the same time, because their knowledge is perhaps complementary. Coopetition is the combination of cooperation and competition. When two entities are coopetiting they compete within a single organization to provide a solution for a specific adaptation state. Nevertheless, they share their knowledge in order to achieve a better solution.
5. *Competition*: in this case at least two entities compete to carry out a task. There is no knowledge sharing among the competitors, since each entity aims at winning the competition by defeating the rest of the entities.

The design and implementation of a system with some adaptation capabilities can be tackled in different ways. Nevertheless, multi-agent systems present some qualities that can foster a more natural design of the adaptation capabilities. From the adaptation stages previously discussed, we can infer that some reasoning capabilities are usually required. i.e., in *Initiative* stage for an adaptive system, if the system has to detect a need for adaptation, there are two main choices: 1) work as a reactive system, where given a stimulus the system reacts, or 2) carry out an inference process where stimuli and stored information are used. Another stage in an adaptation life-cycle, where reasoning capabilities are especially interesting, is the *Specification of Adaptation*. In this stage the system has to decide which

adaptation will be applied. Therefore, reasoning about what adaptation is better would be desirable. Another motivation for using MAS in adaptation is extensibility. MAS can be easily extended to include new functionalities, i.e. add new adaptation stages to the system. Moreover, most software agent-based approaches use the BDI model (Beliefs, Desires, Intentions) (Bratman, 1987), which is inspired by human reasoning theories. Since supporting GISATIE life-cycle implies managing negotiation, consultancy, delegation, coopetion and competition between the different entities in the adaptation life-cycle (the user, the machine or a third-party). Multi-agent systems are especially suitable, since there is already some work done within agents' research community regarding how the different agents involved in a multi-agent system collaborate or compete by negotiating, delegating or consultancy duties. Lastly, another advantage of the in multi-agent systems is the natural distribution of computation, which supports the integration of the implemented multi-agent system with existing services seamlessly.

3.1. Goals for Adaptation

The goals that ensure adaptation of UI may be established, maintained and updated by any of the entities involved (user, machine, or a third-party). Although the main benefit of adaptation aims at the end user, different aspects of the context of use can also be considered, such as: the user profile, task(s), the computational platform (both hardware and software), and the entire physical and organizational environment in which the task is performed. The goals can be classified as: self-expressed or machine-expressed (locally or remotely), according to their location: the user's mind (U), a local machine (M), or a remote machine (T). A typical example of machine-expressed goals is encountered when the machine is in charge of maintaining a certain level of fault-tolerance depending on varying network or hardware conditions. This main goal can be further decomposed into sub-goals, like keeping a minimal amount of information, avoiding any task disruption.

An adaptation goal expresses *what* is adapted, and *to what* is adapted. *What* is adapted in a UI are (Brusilovsky, 2001) the contents shown (i.e., text, videos, images), how these contents are presented (i.e., size, colors, what widgets are used, fonts, layout) and navigation or dialog (i.e., structure of links, menus). The adaptation can be applied to any characteristic considered in the context of use, including, but not limited to, user characteristics, user skills, user knowledge, the software and hardware platform, the physical

environment attributes or the current task the user is carrying out. For instance, adapting the contents of a web page to a PDA platform and/or to maximize its accessibility (to blind users, elder people). Often, adaptation goals are aimed at preserving some constraints, such as, adapting the presentation to minimize loading time or power consumption, according to the lighting conditions or preferred user channel.

The goals for UI adaptation represent the motivation to initiate an adaptation life-cycle. In MAYA, when these goals are in the user's head, our system cannot directly achieve them, however the system supports this process by means of some adaptability facilities implemented. Although, not every user's goal can be supported, including support for some of them already increases user's trust in the adaptation capabilities of the system. When the goals are kept by the machine, they should be expressed in terms of the context of use characteristics considered during the design of the system and the quality criteria to be preserved. Thus, the goals to be stored must make use of context of use characteristics that the machine is able to either query or store and quality criteria to be preserved (i.e., accessibility, performance, fault-tolerance, continuity, etc.). In MAYA, they are expressed as a quality trade-off. This quality trade-off specifies the quality criteria that should be preserved while adapting the UI. For instance, if in the quality trade-off we specify that continuity and accessibility should be maximized, then the machine will always choose those adaptations producing a lesser disruption in continuity and accessibility, unless the user forces the execution of another adaptation. This quality trade-off is expressed by using i* notation (Yu, 1997), which was originally designed to specify system goals in early requirements analysis stage by using a goal-oriented notation.

3.2 Initiative for Adaptation

This second stage describes how the adaptation life-cycle is started. In this context, three scenarios are possible: the adaptation can be initiated by the user (explicit initiative), by the machine (that detects a change in the context of use that requires adaptation) or both jointly (as a decision taken by the entities in control: U, M or T (broker)). One example of the machine and the user collaborating in this stage occurs when the machine initiates the adaptation and the user cancels it. This stage is further refined into formulation for an adaptation request, detection of an adaptation need, and

notification for an adaptation request, depending on their location: respectively, U, M, or T. In MAYA the adaptation life-cycle can be triggered by the user, the machine or a third-party. The user is allowed to do it by clicking on an extra option available in every UI generated by the machine. Auditory UIs are not currently supported. The machine can also decide that an adaptation is needed by inferring it from the incoming information from the context of use. The changes in the context of use are detected by means of sensors. These sensors can be either software or hardware sensors. Hardware sensors are built-in or plugged into the hardware platform where the application is running, while software sensors are programmed, and included into the applications supported by the MAS, i.e., a software sensor to detect idle time could be easily included. Therefore, MAYA supports the initiative stage to be carried out by either the user or the machine.

In MAYA the machine initiates the adaptation as a reaction only to those changes that are significant. The significance level is perceived differently according to the context. For example, a reduction of the screen space available of 20 pixels will probably be not significant enough in a GUI of a PC, but it can be relevant in a smartphone. This significance of the changes in the parameters of the context of use can be modeled by means of fuzzy sets, with the specification of fuzzy rules.

Eisenstein et al. (2000) provide an example of third-party initiative: the UI is specified by means of XIML, which supports the components of the UI to be provided by external agents. This third-party triggers an adaptation given an update in the widget server or according to user preferences to provide a widget more suitable for the current user.

3.3 Specification of Adaptation

The specification stage is composed of two phases: the proposal and the decision. After the adaptation is initiated, a set of proposals is defined and provided. This set may contain none, one or multiple proposals. These proposals can be elaborated by different entities: the user (U), the machine (M) or a third-party (T). If the machine is the provider of adaptation proposals the adaptation rules can be generated by computation, that is, the machine is able to infer new rules from existing ones or other sources of information.

Once the set of proposals is provided, a decision must be made. The decision phase consists of two steps: first the proposals must be analyzed and accepted or rejected, then, in case of acceptance, it is necessary to decide

among the set of proposals provided, which one (or which ones) is the most appropriate to be applied.

One possible alternative to perform the decision consists in informing the machine about the number (or any other identification data) associated with the proposal chosen. This decision can be made by the user, the machine, a third-party, or even by any combination of the three aforementioned entities. In this case, a mechanism is required to define specifically which proposal among the entities' decisions will be finally applied. The mechanism can be implemented with different approaches, such as associating priorities to the entities or allowing them to vote. For example, in a voting mechanism, each entity votes for one or more proposals, and the one with more votes is applied (ties scenarios must be considered).

Another approach to make decisions consists in prioritizing the proposals according to their compliance to pre-defined criteria, regarding for instance, software qualities. In this regard, the QOC (Questions, Options, Criteria), a notation proposed by MacLean et al. in 1991, can be adopted. By using the QOC notation, the adaptations are associated with the criteria they leverage. These associations are graphically represented with positive or negative signs, according to the way the criterion is affected. Figure 1 illustrates the adaptation of the background color regarding legibility, easiness and speed.

In this example the choice in dark blue color affects positively the legibility, negatively the speed and does not interfere with the easiness. This notation provides a basis for a more complex discussion about the potential tradeoffs that may occur between software qualities (MacLean et al., 1991).

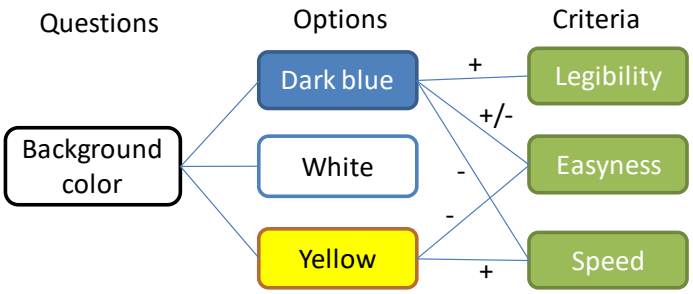

Figure 1. QOC notation: Adaptation of the background color regarding legibility, easiness, and speed.

Regardless of whether the user or the machine have started the adaptation life-cycle, MAYA automatically proposes the set of adaptation rules that best

fit the current context of use. The specification of the pool of available adaptations to choose from is built in different ways. The user can demonstrate how he would like the UI to be adapted (widget colors, sizes, etc.). MAYA supports also the specification of rules by computation, that is, rules created automatically from other rules or from other data sources. It is currently constrained to the refinement of rules previously defined. However, the main corpus of adaptation rules is usually provided by the application designer who defines how the system should react to the different situations resulting from the interaction in specific contexts of use. If the user initiated the adaptation MAYA will ask him which adaptation between the eligible ones he would like to apply. Thus, a UI supporting this task must be delivered to the client. Otherwise, if the user delegates the task of choosing the adaptation to MAYA, it will choose the most appropriate ones, creating a ranking of rules. To make this selection the rules are evaluated by using a set of metrics (López-Jaquero et al., 2008).

3.4 Application of the Adaptation

This stage specifies which entity will apply the adaptation chosen in the previous stage. Since this adaptation is always applied to the UI, the UI should always provide a mechanism that supports adaptation, for instance by provided an API for parametrization (Cockton, 1987). The U adapts (e.g., through UI options, customization, personalization) or the M does it on behalf of the user. For instance, transformations can be applied to apply surface adaptations of a GUI in an information system (Aquino et al., 2010).

Afterwards, MAYA will try to execute the rules starting from the highest one in the ranking of rules previously produced. If the application of the rule does not meet the quality trade-off specified in the goals for UI adaptation, then that rule will be discarded and the MAS will try to apply the next rule in the ranking, following the same process as for the first rule in the ranking. This process is made until no rule is left in the ranking list or until the MAS finds that the score reached in the list is too low for that rule to be applied. The MAS is designed so it will not apply an adaptation rule unless it is considered good enough (unless the user forces its execution). Most of the times, it is better inaction than applying a rule that is not good enough, since it can produce a degradation of UI usability or damage users' trust in the system. One issue found in adaptive systems is that if the adaptations are not properly carried out, and the user starts feeling like he is losing the control of

the application, then the adaptation engine might be easily rejected. Therefore, it is really important for an adaptation architecture to support the user in taking control of the adaptation engine, because mental models and tastes for different users might differ. Therefore it is a good choice to include a mechanism to undo adaptations or to avoid forcing applying an adaptation.

The adaptation can be made at run-time, but also at design-time. For example, a calculator application is adapted at compile-time offering the user a set of options to decide what to include or exclude in the application (Schlee and Vanderdonckt, 2004). Recompiling thr project results into an adapted UI.

3.5 Transition of Adaptation

To improve the continuity in the adaptation life-cycle, a new stage must be included in the adaptation life-cycle. It aims at making smoother the transition from the original UI to the adapted one. This stage is one of the extra stages in GISATIE adaptation framework with respect to Dieterich's one.

This stage specifies which entity will ensure a smooth transition between the original UI and the adapted one. For instance, if M is responsible for this stage, it could provide some visualization techniques, which will present the intermediary steps executed during the adaptation life-cycle, e.g., through animation, morphing or progressive rendering (Rogers and Iba, 2000).

The transition mechanism must be carefully designed, since although it helps in avoiding user disruption, it may have a negative impact in the performance. For instance, when complex animations are used, they can require extensive processing capabilities not always available to the user what can result in degrading the user experience.

Making smoother and clearer the transition between the original UI and the adapted one is very important to avoid confusing the user, and therefore to avoid degrading the users' trust in the system. Although many different kinds of transitions from the original UI to the adapted one can be imagined (Rogers and Iba, 2000), in MAYA we are just supporting those that are general enough to be applied to many different UIs. Besides, our transitions are generated at run-time on-the-fly. MAYA takes the adapted UI and it creates smooth transitions depending on the kind of adaptation the UI has undergone. Right now, it is able to highlight the adapted widgets in different ways to guide the user, by changing the background color of some components, changing the panel containing the adapted components or

adding word balloons to explain the user what happened during the adaptation. Other techniques such as image animation or morphing could be implemented as well.

3.6 Interpretation of Adaptation

This stage specifies which entity will produce meaningful information in order to facilitate the understanding of the adaptation to other entities. Typically, when M performs some adaptation without any explanation, U does not necessarily understand why this type of adaptation has been performed. Conversely, when U performs some adaptation, the user should tell the machine how to interpret this adaptation. For instance, Eisenstein et al. (2000) developed a machine-learning algorithm where the machine first proposes some adaptation to be applied. If this adaptation does not correspond to the user needs, the user is provided with an alternative adaptation and informs the machine how to incorporate this new scheme for future use. The machine updates the knowledge base by interpreting this explanation.

In MAYA, if the user is the one responsible for this stage in the adaptation life-cycle, he is allowed to provide a description of what the adaptation was useful for. It allows the machine to extract some keywords used to relate the new adaptation to other adaptations stored in the adaptation rules repository. On the other hand, if the machine is responsible for this stage, it always adds a tooltip to the adapted UI with a short description of the adaptation made (this text must be provided by the designer of the adaptation rule). This simple add-in to the generated UI helps the user to understand what happened, so he can better interpret what is going on. Therefore, MAYA supports the transition stage to be carried out by either the machine or the user.

3.7 Evaluation of Adaptation

This stage specifies the entity responsible for evaluating the quality of the adaptation performed so that it will be possible to check whether or not the goals initially specified are met. For instance, if M maintained some internal plan of goals, it should be able to update this plan according to the adaptations applied so far. If the goals are in the users' mind, they could also be evaluated with respect to what has been conducted in the previous stages. In this case, the explanation of the adaptation conducted contributes to updating the goals' status too. Collaboration between M and U could be considered.

MAYA tries to achieve QoA (Quality of Adaptation) (López-Jaquero et al., 2008) by selecting the best adaptation rules to apply at a given application stage. Since it is impossible to foresee every combination of changes in the context of use, the machine can apply a rule that is not good enough, or simply it can apply a rule that the user dislikes. Thus, in the system, the user can undo any adaptation applied expressing that he did not like it. This feedback from the user is injected into the adaptation evaluation mechanism by applying a Bayesian approach in which rules can improve or worsen their scores. That is, even though an adaptation rule can obtain a high score initially at application stage, its position in the ranking can be decreased if the user dislikes the adaptation and he informs this by undoing the adaptation rule whenever it is applied. Therefore, the evaluation stage is carried out by the user and the machine in collaboration.

ADAMOS (Arhippainen, 2004) is a project that provides a practical view of user experience studies and methods. Adamos provides a mechanism that enables to evaluate the quality of the adaptation performed. It is done in a way that makes it possible to check whether the adaptation goals initially specified are met or not. For instance, an automated evaluation of the adaptation will be recorded in a knowledge base to be benchmarked with other adaptation rules and strategies. The final outcome of the evaluation of an adaptation will consist in a *feedback loop* that will inform the machine about the success or the failure of the adaptation so as to improve any future adaptation (see Figure 2). Less intrusive techniques can also be imagined. The availability of emotion recognition techniques (Cowie et al., 2001) can provide a mean to automatically assess the satisfaction of the user with the adaptations without disturbing him.

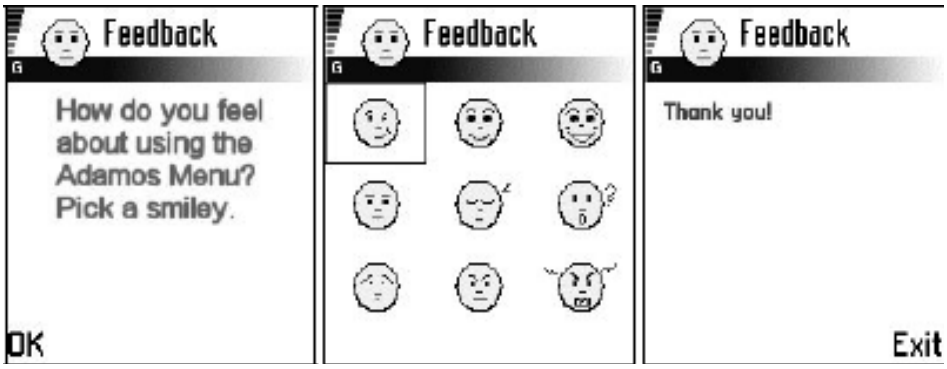

Figure 2. Feedback gathered from the user after adaptation in the ADAMOS project.

4. Conclusion

Adaptation is a complex topic. When dealing with such a complex topic, the availability of methods, processes, guidelines, etc. helps the designer to create better system with adaptation capabilities. In this paper we aim at contributing in this direction. First, an adaptation framework has been thoroughly described, including all seven stages. For each stage a definition, together with some examples and a discussion is provided. A multi-agent system has been used as a running example throughout the paper to better illustrate GISATIE. Moreover, different stakeholders are considered and described, namely: the user, the machine and a third party. An interesting aspect discussed in GISATIE is how these entities can interact to perform a stage in the adaptation life-cycle. Thus, the entities can interact by using negotiation, consultancy, delegation, coopetition or competition. One of the strong points of GISATIE with respect to other adaptation frameworks is supporting more than just the execution part of the adaptation life-cycle, that is to say, covering all the stages related to the evaluation of the adaptation. The evaluation part of the adaptation life-cycle should be a must for any adaptation designed, since it will provide better adaptations with smooth transitions from the original UI to the adapted one, an understanding of why the adaptation took place and an assessment of how good the adaptation actually was. By designing all the seven stages, adaptations will be more easily understood by the user, reducing the chances that the adaptation applied is rejected.

Acknowledgements

This paper is part of the R+D+i project 2gether (PID2019-108915RB-I00) funded by MCIN/AEI/ 10.13039/501100011033.

References

Abrahão, S., Insfrán, E., Sluÿters, A., Vanderdonckt, J.: 2021. Model-based intelligent user interface adaptation: challenges and future directions. *Softw. Syst. Model*. 20(5), 1335-1349. DOI: https://doi.org/10.1007/s10270-021-00909-7

Aquino, N., Vanderdonckt, J., Pastor, O.: 2010. *Transformation templates: adding flexibility to model-driven engineering of user interfaces*. In: Proc. of ACM Symposium on Applied Computing SAC'2010, ACM Press, New York, pp. 1195–1202. DOI: https://doi.org/10.1145/1774088.1774340

Arhippainen, L., Rantakokko, T. and Tähti, M.: 2005. *Navigation with an Adaptive Mobile Map-Application: User Experiences of Gesture- and Context-Sensitiveness*. In: Proc. of 2nd Int. Symposium on Ubiquitous Computing Systems, UCS'2004, Tokyo, November 8-9, 2004. Vol. 3598 of Lecture Notes in Computer Science, Springer, Berlin, pp. 62–73. DOI: https://doi.org/10.1007/11526858_6

Benaboud, R. and Sahnoun, Z. 2006. *Implémentation par agents du processus d'adaptation des informations en interface homme-machine*. In: Proc. of the 18th Int. Conf. of the Association Francophone d'Interaction Homme-Machine IHM '06. ACM Press, New York, NY, USA, pp. 273–276. DOI: https://doi.org/10.1145/1132736.1132782

Benyon, D. and Murray, D. 1993. *Developing adaptive systems to fit individual aptitudes*. In: Proc. of the 1st International Conference on Intelligent User Interfaces IUI 1993, Orlando, Florida. ACM Press, New York, NY, pp. 115-121. DOI: https://doi.org/10.1145/169891.169925

Bratman, M.E.: 1987. *Intention, Plans, and Practical Reason*. Harvard University Press, Cambridge.

Browne, D.P., Sharratt, B. and Norman, M.A.: 1986. The formal specification of adaptive user interfaces using Command Language Grammar. In: Proc. of the ACM Conference on Human Aspects in Computing Systems CHI'1986, Boston, April 13-17, 1986. ACM Press, New York, pp. 256–260. DOI: https://doi.org/10.1145/22339.22381

Brusilovsky, P.: 2001. Adaptive Hypermedia, *User Modeling and User-Adapted Interaction* 11, 1-2, pp .87–110.

Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J.: 2003, A Unifying Reference Framework for multi-target user interfaces. *Interact. Comput*. 15(3), pp. 289-308. DOI: https://doi.org/10.1016/S0953-5438(03)00010-9

Cockton, G.: 1987. Some Critical Remarks on Abstractions for Adaptable Dialogue Managers. In: Proc. of BCS HCI 1987: 325-343.

Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., &

Taylor, J. G.: 2001. Emotion recognition in human-computer interaction. *Signal Processing Magazine*, IEEE, 18(1), 32-80. DOI: https://doi.org/10.1109/79.911197

Dessart, Ch.-E., Genaro Motti, V., Vanderdonckt, J.: 2012. Animated transitions between user interface views. In: Proc. of ACM Conf. On Advanced Visual Interfaces AVI 2012. ACM Press, New York, pp. 341-348. DOI: https://doi.org/10.1145/2254556.2254623

Dieterich, H., Malinowski, U., Kühme, T.and Schneider-Hufschmidt, M.: 1993, State of the Art in Adaptive User Interfaces. In: Schneider-Hufschmidt, M., Khüme, T., Malinowski, U. (Eds.), *Adaptive User Interfaces: Principle and Practice*. North Holland, Amsterdam, pp. 13-48.

Eisenstein, J., Vanderdonckt, J., and Puerta, A.R.: 2000. Adapting to mobile contexts with user-interface modeling. In: Proc. of the Third IEEE Workshop on Mobile Computing Systems and Applications WMCSA 2000, IEEE Press, Los Alamitos, pp. 83-92. DOI: https://doi.org/10.1109/MCSA.2000.895384

Furtado, E., Furtado, V., Silva, W.B., Rodrigues, D.W.T., da Silva Taddeo, L., Limbourg, Q., Vanderdonckt, J.: An ontology-based method for designing multiple user interfaces. In: Proceedings of International Workshop on Multiple User Interfaces, MUI' 01 (2001). https://www.researchgate.net/publication/2567741_An_Ontology-Based_Method_for_Universal_Design_of_User_Interfaces

Gardiner, M. and Christie, B.: 1987. *Applying Cognitive Psychology to User Interface Design*. John Wiley, New York.

Gómez, J.M. and Tran, T.: 2009. A Survey on Approaches to Adaptation on the Web. In: Lytras, M.D., Ordóñez de Pablos, P. (Eds.), *Emerging Topics and Technologies in Information Systems.* IGI Global Publishing, Hershey, pp. 136-152. DOI: 10.4018/978-1-60566-222-0.ch007

Horvitz, E.: 1999. *Principles of Mixed-Initiative User Interfaces*. In: Proceedings of ACM Conference on Human Factors in Computing Systems CHI'99, Pittsburgh, May 15-20, 1999. ACM Press, New York, pp. 159–166. https://doi.org/10.1145/302979.303030

Huhtala, J., Mäntyjärvi, J., Ahtinen, A., Ventä, L. and Isomursu, M.: 2009. *Animated Transitions for Adaptive Small Size Mobile Menus*. In: Proc. of the 12th IFIP TC 13 Int. Conference on Human-Computer Interaction, Interact'2009, Uppsala, August 24-28, 2009. Vol. 5726 of Lecture Notes in Computer Science. Springer, Berlin, pp. 772-781.

Huhtala, J., Sarjanoja, A.-H., Mäntyjärvi, J., Isomursu, M. and Häkkilä, J.: 2010. *Animated UI transitions and perception of time: a user study on animated effects on a mobile screen*. In: Proc. of ACM Conf. on Human Aspects in Computing Systems, CHI'2010, Atlanta, April 10-15, 2010. ACM Press, New York, pp. 1339–1342. DOI: https://doi.org/10.1145/1753326.1753527

Langley, P.: 1999. *User Modeling in Adaptive Interfaces*. In: Kay, J. (Ed.), Proceedings of the 7th International Conference on User Modeling UM'1999, Banff, July 1999. pp. 357-370, Springer, Berlin.

Lavie, T., Meyer, J.: 2010. Benefits and costs of adaptive user interfaces. *International Journal of Human-Computer Studies* 68, pp. 508–524. DOI: https://doi.org/10.1016/j.ijhcs.2010.01.004

Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L., and Florins, M. 2004. *USIXML: A User Interface Description Language Supporting Multiple Levels of Independence*. In: Proc. of Workshops in connection with the 4th Int. Conf. on Web Engineering (ICWE '04). Engineering Advanced Web Applications (28-30 July, 2004), DIWE '04, Matera, M. and Comai, S. (Eds.), Rinton Press, pp. 325–338.

López-Jaquero, V., Vanderdonckt, J., Montero, F., and González, P.: 2007. *Towards an Extended Model of User Interface Adaptation: the ISATINE framework*. In: Proc. of IFIP WG2.7/13.4 10th Conference on Engineering Human Com-puter Interaction jointly organized with IFIP WG 13.2 1st Conference on Human Centred Software Engineering and DSVIS - 14th Conference on Design Specification and Verification of Interactive Systems, EIS'2007, Salamanca, March 22-24, 2007, J. Gulliksen, M.B. Harning, Ph. Palanque (Eds.). Vol. 4940 of Lecture Notes in Computer Science. Springer, Berlin, pp. 374–392. DOI: https://doi.org/10.1007/978-3-540-92698-6_23

López Jaquero, V., Montero, F. and González, P.: 2008. Quality of Adaptation: User Cognitive Models in Adaptation Quality Assesment. In: Proc. of 6th International Conference on Computer-Aided Design of User Interfaces, CADUI'2008, Albacete, June 11-13, 2008. López Jaquero, V., Montero, F., Molina, J.P., Vanderdonckt, J. (Eds.)., Springer, Berlin, pp. 265–276. DOI: https://doi.org/10.1007/978-1-84882-206-1_24

López-Jaquero, V., Montero, F. and Gonzalez, P.: 2009. AB-HCI: an interface multi-agent system to support human-centred computing. *IET Software* 3, 14. pp. 14–25. DOI: https://doi.org/10.1049/iet-sen:20070108

McKinley, P. K., Sadjadi, S. M., Kasten, E. P., and Cheng, B.H.: 2004. Composing Adaptive Software. *IEEE Computer* 37, 7. pp. 56–64. DOI: https://doi.org/10.1109/MC.2004.48

Motti, V.G., Vanderdonckt, J.: 2013. *A computational framework for context-aware adaptation of user interfaces*. In: Proceedings of the 7th IEEE International Conference on Research Challenges in Information Science, RCIS '13, pp. 1–12 (2013). https://doi.org/10.1109/RCIS.2013.6577709

MacLean, A., Young, R.M., Bellotti, V. and Moran, T.P.: 1991. Questions, options, and criteria: elements of design space analysis. Hum.-Comput. Interact. 6, 3 (September 1991), 201-250. DOI: http://dx.doi.org/10.1207/s15327051hci0603\&4_2

Norcio, A.F., Stanley, J.: 1989. Adaptive human-computer interfaces: a literature survey and perspective. *IEEE Transactions on Systems, Man, and Cybernetics*, Volume 19, Issue 2, Mar/Apr 1989, 399-408. DOI: https://doi.org/10.1109/21.31042

Norman, D.A.:1986, Cognitive Engineering. In: Norman, D.A., Draper, S.W. (Eds.): User Centered System Design. Lawrence Erlbaum Associates, Hillsdale, pp. 31–61.

Oppermann, R.: 1994. Adaptively supported adaptability. *International Journal of Human-Computer Studies* 40, 3. pp. 455–472.

Paramythis, A.,Weibelzahl, S., Masthoff, J.: 2010. Layered evaluation of interactive adaptive systems: framework and formative methods. *User Model. User Adapt. Interact*. 20(5), 383–453 (2010). https://doi.org/10.1007/s11257-010-9082-4

Rogers, S. and Iba, W.: 2000, Adaptive User Interfaces: Papers from the 2000 AAAI

Symposium. AAAI Press, Technical Report SS-00-01.

Rosaci, R., Sarnè, G.M.L.: 2006. MASHA: A multi-agent system handling user and device adaptivity of Web sites. *User Modeling and User-Adapted Interaction* 16, 5. pp. 435–462. DOI: https://doi.org/10.1007/s11257-006-9015-4

Schlee, M. and Vanderdonckt, J.: 2004. *Generative Programming of Graphical User Interfaces*. In: Proceedings of 7th Interna-tional Working Conference on Advanced Visual Interfaces, AVI'2004, Gallipoli, May 25-28, 2004. pp. 403-406. ACM Press, New York. DOI: https://doi.org/10.1145/989863.989936

Schlienger, C., Conversy, S., Chatty, S., Anquetil, M. and Mertz, Ch.: 2007. *Improving Users' Comprehension of Changes with Animation and Sound: An Empirical Assessment*. In: Proceedings of IFIP TC 13 Conference on Human-Computer Interaction, Interact'2007, Rio de Janeiro, September 10-14, 2007. Vol. 4662 of Lecture Notes in Computer Science. pp. 207–220, Springer, Berlin. DOI: https://doi.org/10.1007/978-3-540-74796-3_20

Sherman, R., Alpert, J.K., Karat, C., Carolyn, B. and Vergo, J.: 2003. User attitudes regarding a user-adaptive e-commerce web site. *User Modeling and User-adapted Interaction* 13, 4. pp. 373–396. DOI: https://doi.org/10.1023/A:1026201108015

Sendín, M., López-Jaquero, V. and Collazos, C.A.: 2008. Collaborative Explicit Plasticity Framework: a Conceptual Scheme for the Generation of Plastic and Group-Aware User Interfaces. *Journal of Universal Computer Science* 14, 9. pp. 1447–1462. DOI: http://dx.doi.org/ 10.3217/jucs-014-09-1447

Totterdell, P. and Rautenbach, P.: 1990. Adaptation as a Problem Design. In: Browne, D., Totterdell, P., Norman, M. (Eds.), *Adaptive User Interfaces*. Computer And People Series, Acadamic Press, pp. 59–84.

Yu, E.: 1997. *Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering*. In: Proceedings of the 3rd IEEE International Symposium on Requirements Engineering, RE'97, Washington, January 6-8, 1997. pp. 226-235, IEEE Computer Society Press, Los Alamitos. DOI: http://dx.doi.org/10.1109/ISRE.1997.566873.

.