Model-based User-Interface Adaptation by Exploiting Situations, Emotions and Software Patterns

Christian Märtin[1], Christian Herdin[1,2] and Jürgen Engel[1,2]
[1]Faculty of Computer Science, Augsburg University of Applied Sciences, Augsburg, Germany
[2]Institute of Computer Science, University of Rostock, Rostock, Germany

Keywords: Situation Analytics, Emotion Recognition, Adaptive User Interfaces, HCI-Patterns, Model-based User Interface Design.

Abstract: This paper introduces the SitAdapt architecture for adaptive interactive systems that are situation-aware and respond to changing contexts, environments and user emotions. An observer component is watching the user during interaction with the system. The adaptation process is triggered when a given situation changes significantly or a new situation arises. The necessary software modifications are established in real-time by exploiting the resources of the PaMGIS MBUID development framework with software patterns and templates at various levels of abstraction. The resulting interactive systems may serve as target applications for end users or can be used for laboratory-based identification of user personas, optimizing user experience and assessing marketing potential. The approach is currently being evaluated for applications from the e-business domain.

1 INTRODUCTION

Model-based user interface design (MBUID) environments offer a multitude of model categories and tools for building interactive systems for all platforms and device types that meet both functional and user experience requirements. By using models for specifying the business domain, user tasks, user preferences, dialog and presentation styles, as well as device and platform characteristics, the resulting software apps can be tailored to varying user needs and target devices.

Model-based development can also provide for responsiveness, by controling the changing layout and presentation requirements when migrating from one device type (e.g. desktop) to another (e.g. smartphone) in real-time. The CAMELEON reference framework (Calvary et al., 2002), a de-facto standard architecture for MBUIDEs also includes model categories that allow for adapting the target software in pre-defined ways before software generation or during runtime.

However, evolution in software engineering is progressing rapidly and recently has led to the emergence of intelligent applications that monitor the emotional state of the user – together with the changing interaction context – in order to better understand the individual users' behavior and both improve user experience and task accomplishment by letting the software react and adapt to changing individual requirements in real-time.

In other words, the interaction context is analyzed and the users' situation is recognized by such intelligent monitoring systems. The term situation analytics (Chang, 2016) was coined to stimulate the development of sound software engineering approaches for developing and running such situation-aware adaptive systems that ultimately also recognize the mostly hidden mental state of the system users and are able to react to it.

In this paper we present a new approach for building situation-analytic capabilites and real-time adaptive behavior into interactive applications that were designed by MBUID frameworks. We present a prototypical implementation of our runtime-architecture that demonstrates how model-based design can fruitfully be combined with synchronized multi-channel emotion monitoring software that is used to trigger the real-time adaptation of the interactive target application by exploiting pattern and model repositories at runtime.

MĂďrtin C., Herdin C. and Engel J.
Model-based User-Interface Adaptation by Exploiting Situations, Emotions and Software Patterns.
DOI: 10.5220/0006502400500059
In *Proceedings of the International Conference on Computer-Human Interaction Research and Applications (CHIRA 2017)*, pages 50-59
ISBN: 978-989-758-267-7

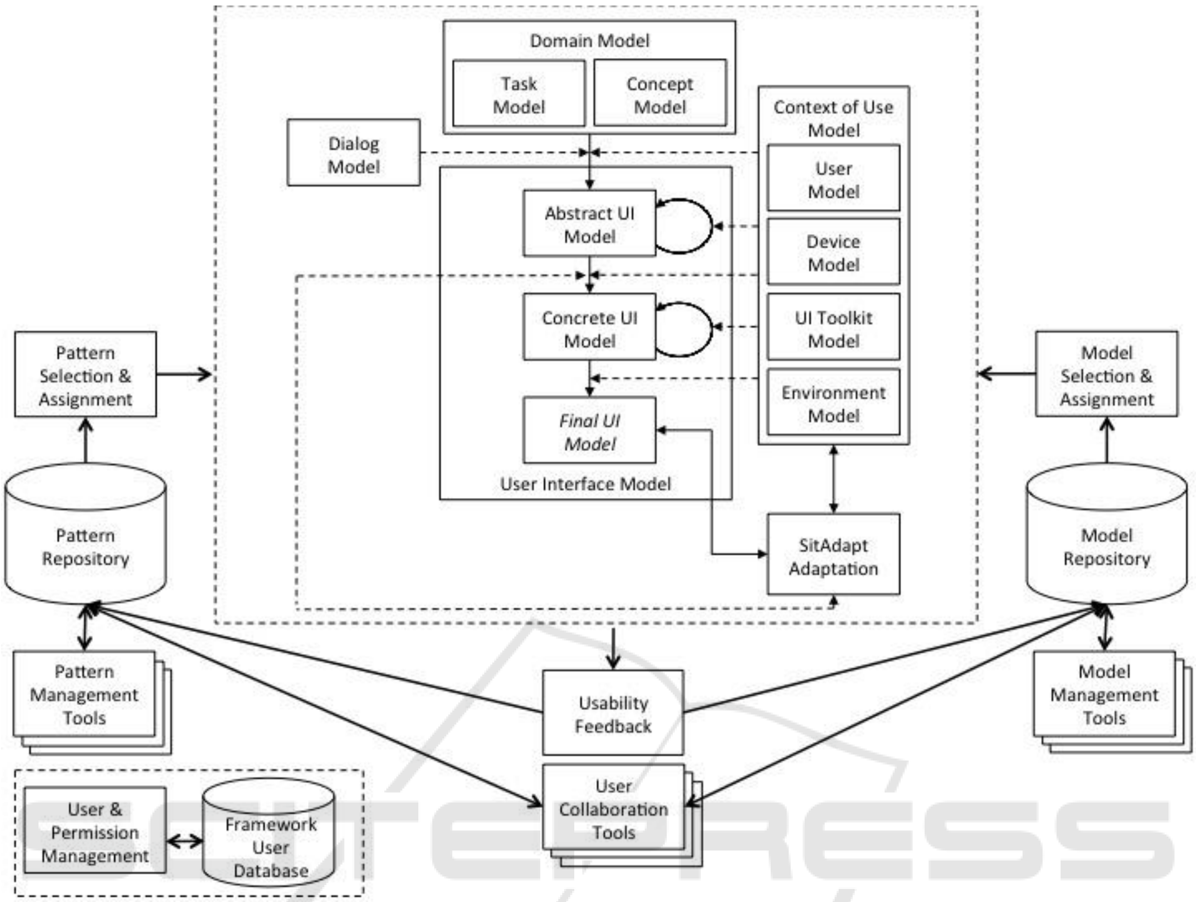

Figure 1: Overview of the PaMGIS development framework with the new SitAdapt component.

The main contributions of this paper are the following:

- Design of architecture extensions for an existing MBUID framework for enabling real-time adaptation of target application user interfaces by using situation analytics
- Discussion of the use of situation and HCI patterns for adaptation purposes
- Detailed discussion of the adaptation process with the focus on user-related adaptations

The paper is organized in the following way. Chapter 2 discusses related work in the field of model- and pattern-based user interface development systems and defines the underlying concept of "situation". The chapter also discusses the related work in the areas of situation-aware systems, model and pattern based construction of interactive systems, and emotion recognition.

Chapter 3 introduces SitAdapt, our architectural approach for developing and running situation-analytic adaptive interactive systems. The chapter focuses on discussing the adaptation process and the various adaptation types supported by the system in detail. The specification language of the HCI patterns and model fragments used in the process is discussed elsewhere (Engel et al. 2015).

Chapter 4 demonstrates the SitAdapt component in action. It gives an example for a situation recognized in a travel-booking application and shows, how the adaptation process would react in order to improve the user experience and task accomplishment for a specific user.

Chapter 5 concludes the paper with a short discussion of currently evolving target applications and gives hints to our planned future work.

2 RELATED WORK

In this paper we present the user interface adaptation process of the SitAdapt system that combines model- and pattern-based approaches for interactive system construction with visual and bio-signal-based emotion recognition technology to allow for software adaptation by real-time situation analytics. The underlying software and monitoring technology was introduced in (Märtin et al., 2016).

2.1 Model- and Pattern-based User Interface Design

Model-based User Interface Design (MBUID) during the last decades has paved the way for many well-structured approaches, development environments and tools for building high-quality interactive systems that fulfil tough requirements regarding user experience, platform-independence, and responsiveness. At the same time model-driven and model-based architectural and development approaches (Meixner et al., 2014) were also introduced to automate parts of the development process and shorten time-to-market for flexible interactive software products. As one de-facto process and architectural standard for MBUID the CAMELEON reference framework has emerged (Calvary et al., 2002). CRF defines many model categories and a comprehensive base architecture for the construction of powerful model-based development environments.

In order to allow for automation of the development process for interactive systems, to raise the quality and user experience levels of the resulting application software and to stimulate developer creativity, many software-pattern-based design approaches were introduced during the last 15 years (Breiner et al., 2010).

In order to get the best results from both, model and pattern-based approaches, SitAdapt is integrated into the PaMGIS (Pattern-Based Modeling and Generation of Interactive Systems) development framework (Engel et al., 2015), which is based on the CAMELEON reference framework (CRF). PaMGIS contains all models proposed by CRF, but also exploits pattern-collections for modeling and code generation. The CRF is guiding the developer on how to transform an abstract user interface over intermediate model artifacts into a final user interface. The overall structure of the PaMGIS framework with its tools and resources, the incorporated CRF models, and the new SitAdapt component is shown in figure 1.

The SitAdapt component has full access to all sub-models of the context of use model and interacts with the user interface model.

Within CRF-conforming systems the abstract user interface model (AUI) is generated from the information contained in the domain model of the application that includes a task model and a concept model (i.e. typically an object-oriented class model defining business objects and their relations) and defines abstract user interface objects that are still independent of the context of use.

The AUI model can then be transformed into a concrete user interface model (CUI), which already exploits the context model and the dialog model, which is responsible for the dynamic user interface behavior.

In the next step PaMGIS automatically generates the final user interface model (FUI) by parsing the CUI model. To produce the actual user interface, the resulting XML-conforming UI-specification can either be compiled or interpreted during runtime, depending on the target platform. Chapter 3 discusses in detail, how SitAdapt complements and accesses the PaMGIS framework.

2.2 Context- and Situation-Awareness

Since the advent of smart mobile devices, HCI research has started to take into account the various new usability requirements of application software running on smaller devices with touch-screen or speech interaction or of apps that migrate smoothly from one device type to another. Several of the needed characteristics for these apps targeted at different platforms and devices can be specified and implemented using the models and patterns residing in advanced MBUID systems. Even runtime support for responsiveness with the interactive parts distributed or migrating from one (virtual) machine to the other and the domain objects residing in the Cloud can be modeled and managed by CRF-conforming development environments (e.g. Melchior et al., 2011).

The concept of context-aware computing was originally proposed for distributed mobile computing in (Schilit et al., 1994). In addition to software and communication challenges to solve when dynamically migrating an application to various devices and locations within a distributed environment, the definition of context given then also included aspects such as lighting or the noise level as well as the current social situation (e.g. are other people around?, are these your peers?, is one of them your manager?, etc.).

Since then, mobile software has made huge steps towards understanding of and reacting to varying situations. To capture the individual requirements of a situation (Chang, 2016) proposed that a situation consists of an environmental context *E* that covers the user's operational environment, a behavioral context *B* that covers the user's social behavior by interpreting his or her actions, and a hidden context M that includes the users' mental states and emotions. A situation *Sit* at a given time t can thus be defined as $Sit = \langle M, B, E \rangle_t$. A user's intention for using a specific software service for reaching a goal can then be formulated as temporal sequence $\langle Sit_1, Sit_2, \ldots, Sit_n \rangle$, where Sit_1 is the situation that triggers the usage of a service and Sit_n is the goal-satisfying situation. In (Chang et al., 2009) the Situ framework is presented that allows the situation-based inference of the hidden mental states of users for detecting the users intentions and identifying their goals. The framework can be used for modeling and implementing applications that are situation aware and adapt themselves to the users' changing needs over runtime.

Our own work, described in the following chapters, was inspired by Situ, but puts most emphasis on maintaining the model-based approach of the PaMGIS framework by linking the domain and user interface models with the user-centric situation-aware adaptation component.

An approach for enabling rich personal analytics for mobile applications by defining a specific architectural layer between the mobile apps and the mobile OS platform is proposed in (Lee, Balan, 2014). The new layer allows to access all sensors and low-level I/O-features of the used devices.

The users' reactions to being confronted with the results of a hyper-personal analytics system and the consequences for sharing such information and for privacy are discussed in (Warshaw et al., 2015).

For implementing the emotion recognition functionality that can be exploited for inferring the desires and sentiments of individual users while working with the interactive application, the current version of SitAdapt captures both visual and biometric data signals. In (Herdin et al., 2017) we discuss the interplay of the various recognition approaches used in our system.

In (Picard, 2014) an overview of the potential of various emotion recognition technologies in the field of affective computing is given. In (Schmidt, 2016) emotion recognition technologies based on bio-signals are discussed, whereas (Qu, Wang, 2016) discusses the evolution of visual emotion recognition approaches including the interpretation of facial macro- and micro-expressions.

2.3 Context-Adaptation

The recognition of emotions and the inference of sentiments and mental states from emotional and other data is the basis for suggesting adaptive changes of the content, behavior, and the user interface of the target application. In general, three different types of adaptation can be distinguished when focusing on user interfaces (Akiki et al., 2014), (Yigitbas et al., 2015):

- *Adaptable user interfaces.* The user interface is a-priori customized to the personal preferences of the user.
- *Semi-automated adaptive user interfaces*. The user interface provides recommendations for adaptations, which can be accepted by the user or not.
- *Automated adaptive user interfaces.* The user interface automatically reacts to changes in the context of the interactive application

The SitAdapt runtime-architecture co-operates with the PaMGIS framework to support both, semi-automated and automated user interface adaptation. For constructing real situation-aware systems, however, user interface adaptation aspects have to be mixed with content-related static and dynamic system aspects that are typically covered by the task and context model, both together forming the domain model of the framework.

In (Mens et al., 2016) a taxonomy for classifying and comparing the key-concepts of diverse approaches for implementing context-awareness is presented. The paper gives a good overview of the current state-of-the-art in this field.

3 SITADAPT – ARCHITECTURE

In order to profit from earlier results in the field of MBUID systems, the SitAdapt runtime environment is integrated into the PaMGIS (Pattern-Based Modeling and Generation of Interactive Systems) development framework. The SitAdapt environment extends the PaMGIS framework and allows for modeling context changes in the user interface in real-time.

The architecture (fig. 2) consists of the following parts:

- The *data interfaces* from the different devices (eye-tracker, wristband, FaceReader software and metadata from the application)
- The *signal synchronization component* that synchronizes the data streams from the different input devices by using timestamps
- The *recording component* that records the eye- and gaze-tracking signal of the user and tracks his or her emotional video facial expression

with the Noldus FaceReader software as a combination of the six basic emotions (happy, sad, scared, disgusted, surprised, and angry). Other recorded data about the user are, e.g., age-range and gender. The stress-level and other biometric data are recorded in real-time by a wristband. In addition, application metadata are provided by the target application.

The *situation analytics component* analyzes the situation by exploiting the data and a status flag from the FUI or AUI in the user interface model. A situation profile of the current user is generated as a dynamic sequence of situations.

The *decision component* uses the data that are permanently provided by the situation analytics component to decide whether an adaptation of the user interface is currently meaningful and necessary. Whether an adaptation is meaningful depends on the predefined purpose of the situation-aware target application. Goals to meet can range from successful marketing activities in e-business, e.g. having the user buying an item from the e-shop or letting her or him browse through the latest special offers, to improved user experience levels, or to meeting user desires defined by the hidden mental states of the user. Such goals can be detected, if one or more situations in the situation profile trigger an application dependent or domain independent situation pattern. Situation patterns are located in the pattern repository. If the decision component decides that an adaptation is necessary, it has to provide the artifacts from the PaMGIS pattern and model repositories to allow for the modification of the target application by the adaptation component. The situation pattern provides the links and control information for accessing and composing HCI-patterns and model fragments, necessary for constructing the modifications.

The adaptation component finally generates the necessary modifications of the interactive target application.

These architectural components provided by the SitAdapt system are necessary for enabling the PaMGIS framework to support automated adaptive user interfaces.

3.1 User Interface Construction

The *domain model* (see fig. 1) serves as the starting point of the process that is used for user interface modeling and generation. The model consists of two sub-models, the *task model* and the *concept model*. The task model is specified using the *ConcurTaskTree (CTT)* notation (Paternò, 2001) as well as an XML file, the *UI configuration file*, generated from the CTT description and from accessing contents of the context of use model (see fig. 1).

The *context of use model* holds four sub-models: the user model, the device model, the UI toolkit model, and the environment model. All models play important roles in the user interface construction process and are exploited when modeling responsiveness and adapting the target application to specific device types and platforms. For the situation-aware adaptation process, however, the user model is the most relevant of these sub-models.

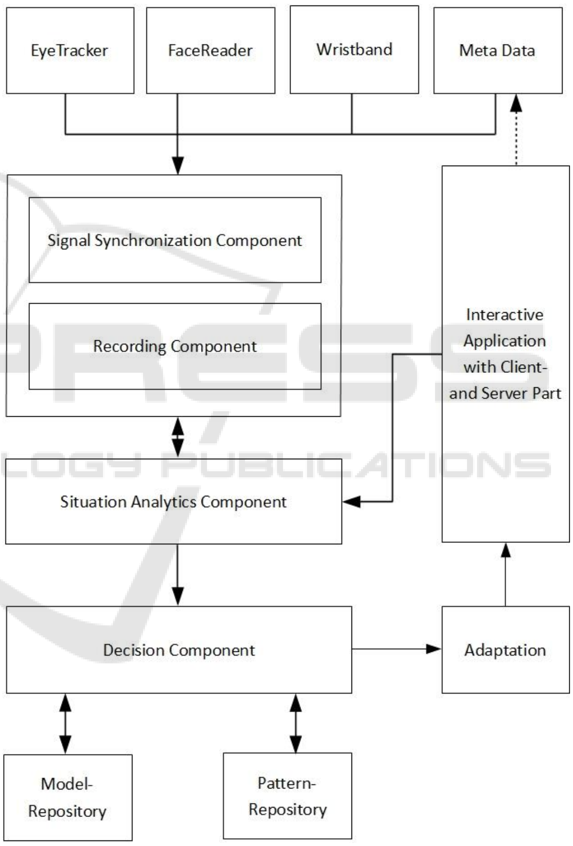

Figure 2: SitAdapt Architecture.

It is structured as follows:

```
<UserCharacteristics>
    <UserIdentData>
    <UserAbilities>
        <USRUA_Visual>
        <USRUA_Acoustic>
        <USRUA_Motor>
        <USRUA_Mental>
    <UserExperiences>
        <USRUE_Domain>
        <USRUE_Handling>
```

```
<UserDistraction>
<UserLegalCapacity>
<UserEmotionalState>
<UserBiometricState>
    <USRBS_Pulse>
    <USRBS_BloodPressure>
    <USRBS_PulseChangeRate>
    <USRBS_E4_DataSet>
    …
    <USRBS_DeviceX_DataSet>
```

The *user model* holds both static information about the current user, and dynamic data describing the emotional state as well as the biometric state. Not all of the attributes need to be filled with concrete data values. The dynamic values concerning emotional state and biometric state are taken from the *situation profile,* whenever an adaptation decision has to be made (see chapter 3.2). Note, however, that the situation profile that is generated by the situation analytics component contains the whole sequence of emotional and biometric user states from session start until session termination. The temporal granularity of the situation profile is variable, starting from fractions of a second. It depends on the target application's requirements.

A priori information can be exploited for tailoring the target application, when modeling and designing the appearance and behavior of the user interface, before generating it for the first time and use. This can already be seen as part of the adaptation process.

Typical a priori data can be user identification data, data about the various abilities of the user, and specific data about the users fluency with the target application's domain and its handling.

Dynamic data will change over time and can be exploited for adapting the user interface at runtime. Such data includes the emotional and biometric state, observed and measured by the hard- and software devices attached to the recording component.

The data structure also allows to directly integrate proprietary data formats provided by the devices used in SitAdapt such as the Empatica E4 wrist band.

If not available a priori, some of the attributes can be completed by SitAdapt, after an attribute value was recognized by the system. For instance the age of a user can be determined with good accuracy by the FaceReader software. With this information, the <UserLegalCapacity> attribute can be automatically filled in.

The generated *UI configuration file* contains a <ContextOfUse> tag field for each task. It has sub-tags that may serve as context variables that hold information relevant for controling the UI configuration and later, at runtime, the adaptation process.

One of the sub-tags may for instance hold information that a task "ticket sale" is only authorized for users from age 18. When the situation analytics component at runtime discovers that the current user is less than 18 years old, a hint is given in the final user interface (FUI) model that she or he is not authorized to buy a ticket, because of her or his age.

The *concept model* consists of the high-level specifications of all the data elements and interaction objects from the underlying business model that are relevant for the user interface. It can therefore be seen as an interface between the business core models of an application and the user interface models. It can, e.g., be modeled using UML class diagram notation. From the concept model there is also derived an XML specification.

In PaMGIS the *task model* serves as the primary basis for constructing the *dialog model* (see figure 1). The various dialogs are derived from the tasks of the task model. Additional input for modeling data types and inter-class-relations is provided by the *concept model*. The dialog model is implemented by using Dialog Graphs (Forbrig, Reichart, 2007).

In the next step the *abstract user interface (AUI)* is constructed by using the input of the domain model and the dialog model. The dialog model provides the fundamental input for AUI specification, because it is based on the user tasks. Each of the different dialogs is denoted as a <Cluster>-element. All elements together compose a <Cluster> that is also specified in XML.

For transforming the AUI into *concrete user interface (CUI)* the AUI elements are mapped to <Form> CUI elements. The context of use model is exploited during this transformation. For instance by accessing the UI toolkit sub-model it can be guaranteed that only such widget types are used in the CUI that come with the used toolkit and for which the fitting code can later be generated in conjunction with the target programming or markup language used for the final user interface.

The CUI specification is also written in XML.

Finally, the CUI specification has to be transformed into the *final user interface (FUI).* The XML specification is therefore parsed and translated into the target language, e.g., HTML, C# or Java.

3.2 User Interface Adaptation Process

For modeling a situation-aware adaptive target application, SitAdapt components are involved in several parts of the entire adaptation process.

Already before the final user interface is generated and displayed for the first time, the situation can be analyzed by SitAdapt. SitAdapt can access the user attributes in the profiles. The situation analytics component gets the synchronized monitored data streams from the recording component and stores the data as the first situation of a sequence of situations in the situation profile.

Thus, the user will get an adapted version of the user interface with its first display, but will not notice that an adaptation has already occurred.

At runtime a situation is stored after each defined time interval. In addition, environmental data, e.g. time of day, room lighting, number of people near the user, age and gender of the user, emotional level, stress level, etc. can be recorded and pre-evaluated and be attached to the situations in the situation profile.

A situation profile has the following structure:

```
<SituationProfile>
 <TargetApplication>
 <User>
 <Situation_0>
     <SituationTime> 0
     <FUI_link> NULL
     <Eye_Tracking> …
     <Gaze_Tracking> …
     <MetaData> NULL
     <Environment>
         <EnvAttrib_1> …
         …
     <EmotionalState> EmoValue_1
     <BiometricState> BioValue_1
 …
 <Situation_i>
     <SituationTime> i
     <FUI_link>FUI_object_x
     <Eye_Tracking> v1,v2
     <Gaze_Tracking> w1,w2.w3
     <MetaData>
         <Mouse> x,y
         …
     <Environment>
         …
     <EmotionalState> angry
     <BiometricState>
         <Pulse>95
         <StressLevel> yellow
         <BloodPressure>160:100
 …
```

This is the structure of the situation profile as currently used for prototypical applications and for evaluating our approach. The attribute <FUI_link> is used to identify the part of the user interface in the focus of the user. It is provided by the eye-tracking data and the mouse coordinates. For more advanced adaptation techniques additional attributes may be required (see chapter 3.3). Situations with their attributes provide a dynamic data base for allowing the decision component to find one or more domain dependent or independent situation patterns in the pattern repository that match an individual situation or a sequential part of the situation profile. The situation patterns hold links to HCI-patterns or model fragments that are used for adaptation purposes.

Typically not all of the existing situation attributes are needed to find a suitable situation pattern.

Also note that some situation attributes are not type-bound. The eye- and gaze-tracking attributes, for instance, can either contain numeric coordinates or sequences of such coordinates to allow for analyzing rapidly changing eye movements in fine-grained situation profiles, e.g. when observing a car driver. However, they can also give already pre-processed application-dependent descriptions of the watched UI objects and the type of eye movements (e.g. a repeating loop between two or more visual objects).

To enable adaptations, some modeling aspects of the PaMGIS framework have to be extended.

When transforming the AUI model into the CUI model the current situation profile of the user is checked by the decision component. The situational data in the profile may match domain independent or application specific situation patterns. If one ore more situation patterns apply, these patterns guide, how a dynamically adapted CUI can be constructed. The construction information is provided by HCI patterns and/or model fragments in the PaMGIS repositories. Each situation pattern is linked with such UI-defining artifacts.

Situation patterns are key resources for the SitAdapt adaptation process. Domain independent situation patterns cover recurring standard situations in the user interface. Application-specific situation patterns have to be defined and added to the pattern repository when modeling a new target application. They mainly cover aspects that concern specific communications between business objects and the user interface. Existing application-specific situation patterns can be reused, if they also apply for a new target application.

The CUI that was modified with respect to the triggering situation then serves as construction template from which the FUI is generated.

However, the FUI is also monitored by SitAdapt at runtime, after it was generated, i.e. when the user interacts with the interactive target application. Thus, the characteristics of the FUI can be modified dynamically by SitAdapt whenever the decision component assesses the occurred situational changes for the user as significant (i.e. the user gets angry or has not moved the mouse for a long time period).

In this case a flag in the UI tells SitAdapt, which part of the AUI is responsible for the recognized situation within a window, dialog, widget and/or the current interaction in the FUI. Depending on the situation analytics results, the detected situation patterns will hint to the available HCI patterns, model fragments and construction resources (e.g. reassuring color screen background, context-aware tool tip, context-aware speech output, etc.). The decision component may thus trigger a modification of the concerned CUI parts.

The adaptation component then accesses and activates the relevant HCI patterns and/or model fragments. From the modified CUI a new version of the FUI is generated and displayed, as soon as possible. After adaptation the FUI is again monitored by the situation analytics component.

3.3 Advanced Adaptation Techniques

Two major goals of adaptive technologies are 1) to raise the effectiveness of task accomplishment and 2) to raise the level of user experience. With the resources, tools, and components available in PaMGIS and SitAdapt we plan to address these goals in the near future.

To monitor task accomplishment, links between the sequence of situations in the situation profile and the tasks in the task model have to be established. For this purpose the situation analytics component needs access to the task model. As the tasks and sub-tasks of the task model are related to business objects, the linking of situations to data objects in the concept model appears to be helpful. With these new communication mechanisms, we can check, whether the sequence of situations goes along with the planned sequence of tasks. If derivations or unforeseen data values occur, situation-aware adaptation can help the user to find back to the intended way.

Both, for situation sequences matching with the task sequences planned by the developer, and for situations that have left the road to the hoped-for business goal, emotional and stress-level monitoring may trigger adaptations of the user interface that raise the joy of use or take pressure from the user in complex situational contexts.

Monitoring different users with SitAdapt while working on the tasks of various target applications in the usability lab can also lead to identifying different personas and usage patterns of the target applications. The findings can also be used for a priori adaptations in the target systems in the case where no situation analytics process is active.

4 SITADAPT IN ACTION

A simple example that demonstrates the functionality of the SitAdapt system for an interactive real-life application is the following e-business case.

A user will book a trip from one city to another on the website fromatob.com. This website applies a wizard HCI-pattern for the booking process. When using SitAdapt for an existing proprietary web-application, a simplified user interface model of the application as well as a task model fragment can be provided in order to make the adaptation functionality partly available for the existing software.

In the first step (fig. 3) the traveller has to enter her or his personal details into the wizard fields.

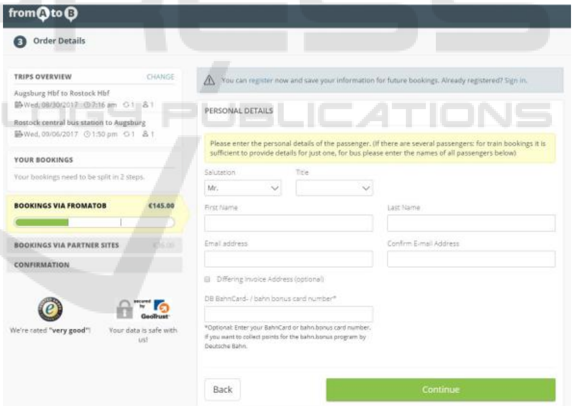

Figure 3: fromatob.com wizard pattern.

SitAdapt is monitoring the user during this task. The SitAdapt system records the eye movements, pulse, stress level, and the emotional state, and gets real-time information from the website (FUI_link).

The situation analytics component creates a user-specific situation from this data:

```
<SituationProfile>
    <TargetApplication>
    <User>
    <Situation_booking>
        <SituationTime>60sec
        <FUI_link>Wizard_Part1
```

```
        <Eye_Tracking>Field
                            Bahncardnumber
        <EmotionalState>angry
        <BiometricState>
            <Pulse>high
            <StressLevel>orange
    </Situation_booking>
```

The Decision Component determines whether an adaptation is necessary with the help of the pattern repository and the model repository. In this example, the component decides that the user has a problem with one field in the form of the wizard (bahncardnumber). The situation can be mapped to the domain-independent situation pattern *form-field-problem*. The form-field-pattern hints to an HCI-pattern from the pattern repository to help the user in this situation. The adaptation creates a new final user interface (FUI) with a chat window in order to help the user (fig. 4).

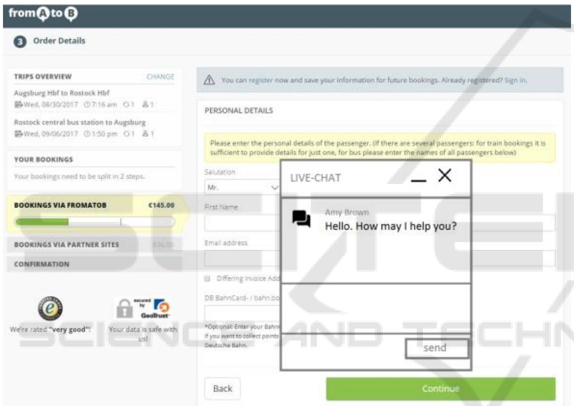

Figure 4: fromatob.com wizard pattern with chat window.

5 CONCLUSION

In this paper we have presented the new adaptive functionality developed for the PaMGIS MBUID framework. This was made possible by integrating a situation-aware adaptation component seamlessly into the model- and pattern-based development environment and by adding runtime-features.

The new SitAdapt component is now fully operational. After finalizing the signal synchronization and recording components, the system is currently being tested and evaluated with target applications from the e-business domain. In one realistic application domain that is currently being implemented we use SitAdapt to watch the user and search for recurring situation patterns in the domain of ticket sale for long-distance-travel. Another area we currently evaluate is the e-business portal of a cosmetics manufacturer. For this domain we make extensive use of usability-lab based user tests with varying scenarios in order to get sufficient data for mining typical application-dependent situation patterns.

For both target application domains we are currently exploring usability, user experience and marketing aspects. It is our next goal to define a large set of situation patterns, both domain-dependent and universally applicable and thus stepwise improve the intelligence level and variety of the resources of the SitAdapt decision component.

REFERENCES

Akiki, P.A., et al.: Integrating adaptive user interface capabilities in enterprise applications. In: Proceedings of the 36th International Conference on Software Engineering (ICSE 2014), pp. 712-723. ACM (2014)

Breiner, K. et al. (Eds.): PEICS: towards HCI patterns into engineering of interactive systems, Proc. PEICS ’10, pp. 1-3, ACM (2010)

Calvary, G., Coutaz, J., Bouillon, L. et al., 2002. “The CAMELEON Reference Framework”. Retrieved August 25, 2016 from http://giove.isti.cnr.it/projects/cameleon/pdf/CAMELEON%20D1.1RefFramework.pdf

Chang, C.K.: Situation Analytics: A Foundation for a New Software Engineering Paradigm, IEEE Computer, Jan. 2016, pp. 24-33

Chang, C.K. et al.: Situ: A Situation-Theoretic Approach to Context-Aware Service Evolution, IEEE Trans. Services Computing, vol. 2, no. 3, 2009, pp.261-275

Engel, J., Märtin, C., Forbrig, P.: A Concerted Model-driven and Pattern-based Framework for Developing User Interfaces of Interactive Ubiquitous Applications, Proc. First Int. Workshop on Large-scale and Model-based Interactive Systems, Duisburg, pp. 35-41, (2015)

Engel, J., Märtin, C., Forbrig. P.: Practical Aspects of Pattern-supported Model-driven User Interface Generation, To appear in Proc. HCII 2017, Springer (2017)

Forbrig, P., Reichart, D., 2007. Spezifikation von „Multiple User Interfaces“ mit Dialoggraphen, Proc. INFORMATIK 2007: Informatik trifft Logistik. Beiträge der 37. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Bremen

Herdin, C., Märtin, C., Forbrig, P.: SitAdapt: An Architecture for Situation-aware Runtime Adaptation of Interactive Systems. To appear in Proc. HCII 2017, Springer (2017)

Lee, Y., Balan, R.K.: The Case for Human-Centric Personal Analytics, Proc. WPA ’14, pp. 25-29, ACM (2014)

Märtin, C., Rashid, S., Herdin, C.: Designing Responsive Interactive Applications by Emotion-Tracking and Pattern-Based Dynamic User Interface Adaptations, Proc. HCII 2016, Vol. III, pp. 28-36, Springer (2016)

Meixner, G., Calvary, G., Coutaz, J.: Introduction to model-based user interfaces. W3C Working Group Note 07 January 2014. http://www.w3.org/TR/mbui-intero/. Accessed 27 May 2015

Melchior, J., Vanderdonckt, J., Van Roy, P.: A Model-Based Approach for Distributed User Interfaces, Proc. EICS '2011, pp. 11-20, ACM (2011)

Mens, K. et al.: A Taxonomy of Context-Aware Software Variability Approaches, Proc. MODULARITY Companion'16, pp. 119-124, ACM (2016)

Moore, R., Lopes, J., 1999. Paper templates. In *TEMPLATE'06, 1st International Conference on Template Production*. SCITEPRESS.

Paternò, F., 2001. ConcurTaskTrees: An Engineered Approach to Model-based Design of Interactive Systems, ISTI-C.N.R., Pisa

Picard, R.:"Recognizing Stress, Engagement, and Positive Emotion", Proc. IUI 2015, March 29-April 1, 2015, Atlanta, GA, USA, pp. 3-4

Qu, F., Wang, S.-J. et al. CAS(ME)2: A Database of Spontaneous Macro-expressions and Micro-expressions, M. Kuroso (Ed.): HCI 2016, Part III, NCS 9733, pp. 48-59, Springer, (2016)

Schilit, B.N., Theimer, M.M.: Disseminating

Active Map Information to Mobile Hosts, IEEE Network, vol. 8, no. 5, pp. 22–32, (1994)

Schmidt, A. Biosignals in Human-Computer Interaction, Interactions Jan-Feb 2016, pp. 76-79, (2016)

Warshaw, J. et al.: Can an Algorithm Know the "Real You"? Understanding People's Reactions to Hyper-personal Analytics Systems, Proc. CHI 2015, pp. 797-806, ACM (2015)

Yigitbas, E., Sauer, S., Engels, G.: A Model-Based Framework for Multi-Adaptive Migratory User Interfaces. In: Proceedings of the HCI 2015, Part II, LNCS 9170, pp. 563-572, Springer (2015)