A Generalized Approach for Context-Aware Adaptation in Mobile E-Learning Settings

Tobias Moebert, Raphael Zender and Ulrike Lucke

Abstract Most existing adaptive tools and applications have been developed specifically for one or few selected scenarios. What is missing is a framework that enables the methodical development of adaptive mobile applications that support a wide variety of educational scenarios. However, challenges are to identify relevant contextual information, to enable the design of adaptive learning scenarios, even by non-technophile teachers, to create a platform-independent way of collecting contextual information and to find a way to satisfy the different demands from universities and industry. To meet some of these challenges, this chapter introduces a systematic approach for developing mobile adaptive learning applications.

Keywords E-learning · Mobile learning · Context detection · Context-awareness · Adaptivity · Framework · Mobile devices

Abbreviations

API	Application programming interface
GUI	Graphical user inferface
ITS	Intelligent Tutoring System
MOTIVATE	Mobile Training Via Adaptive Technologies
OWL	Web Ontology Language
RDF	Resource description framework
RDFS	Resource description framework schema
REST	Representational state transfer
SPARQL	SPARQL protocol and RDF query language

T. Moebert (✉) · R. Zender · U. Lucke
Universität Potsdam, Institut für Informatik, Haus 4, August-Bebel-Straße 89, 14482 Potsdam, Germany
e-mail: tobias.moebert@uni-potsdam.de

R. Zender
e-mail: raphael.zender@uni-potsdam.de

U. Lucke
e-mail: ulrike.lucke@uni-potsdam.de

A. Peña-Ayala (ed.), *Mobile, Ubiquitous, and Pervasive Learning*,
Advances in Intelligent Systems and Computing 406,
DOI 10.1007/978-3-319-26518-6_2

1 Introduction

Today, mobile devices like smart phones and tablet computers are affordable and widely used not only in private domains, but also in business applications and educational efforts. The use of these devices for mobile learning made it possible to access educational material spatially and temporally unbounded. This, in turn, has made it possible to use time periods for learning that were unavailable before. Moreover, learning with mobile devices opened up new possibilities for situated learning, i.e. problem-based learning in real-life situations.

However, mobile devices are offering only a reduced screen size and differ in forms of interaction. In addition, users of mobile learning software tend to have a shorter attention span compared to learning in a formal setting [1]. For this reason, the concept of micro learning, with its focus on relatively small learning units and short-term learning activities, is often preferred. To realize the full potential of micro learning, it is important to offer content to the learner that is on the one hand relevant for their current situation and on the other hand customized to their personal needs. In order to achieve this, contextual information can be used to match educational offers with the learner's individual needs and characteristics. All information that describes the interaction between the user, the application and the environment are considered contextual information [2, 3]. Several research initiatives have identified a plethora of contextual information that might be used for adaptation. Mobile devices offer a variety of sensors and interaction options that can be used to gather contextual information. The challenge is to identify those pieces of contextual information that are considered most relevant by the authors of educational software and can still be reliably detected.

Most existing adaptive tools and applications have been developed specifically for one or few selected scenarios, resulting in some rather specialized prototypes. What yet is missing is a framework that enables the methodical development of adaptive mobile applications that support a wide variety of educational scenarios. Such scenarios could originate from academic as well as business areas. An exemplifying use case for an academic is a biological excursion during which contextual information could be used to lead the students to interesting plants or animals, to help them identify information that is relevant for their learning goals, or to match content with their personal likings.

This would be even more interesting if it was a multi-day excursion. According to [4], a more business-oriented use case would be that of a relief organization that specializes in helping refugees and other people in need. Such an organization operates worldwide and is therefore dependent on a large number of employees as well as volunteers. To train these people, the organization already relies on class lecture, classic e-learning and mobile learning. Mobile learning is especially crucial when it comes to delivering training courses to people who do not have access to class lectures or classical e-learning means. With adaptation mechanisms, the mobile learning system would in future be able to automatically adjust to ambient conditions (e.g. location, noise level, but also temperature and humidity) and to leverage parameters like the user's experience to accurately deliver a suitable learning unit, in an adequate form of media.

In the course of this chapter we want to present a framework for the systematic creation of adaptive mobile applications (Sect. 2) that we developed. We have placed great emphasis on application in both academic and industrial areas, a systematic detection of relevant contextual information (Sect. 3), platform-independence (Sect. 4.2) and finally finding new user experience concepts that ease the use of adaptive applications for authors as well as end users (Sect. 4.3).

2 Adaptive Learning

Since the development of the first teaching machines in 1924, computer-aided learning without a teacher's presence is an important topic in educational research. First attempts followed the theory of behaviorism and simply rated a learner's answers to specific questions [5]. Later, cognitivism-driven research identifies phases of human learning and aimed at their support with the help of intelligent tutoring systems (ITS) [6].

In general, ITS utilize a set of models about domains, learners, and didactical strategies to derive decisions regarding learning process support. The central student model contains the system's beliefs about the learner's knowledge, derived by sources like the history of the learner's behavior [7]. In the last 10–15 years, the trend of mobile devices with a broad spectrum of sensors as well as a permanent internet connection led to a broad spectrum of measurable information about a learner's physical context, social relations and even physical functions, like the heartbeat rate. Educational systems are therefor able to draw on abundant context resources to adapt to learner's as well as teacher's needs, and increase the convenience and efficiency of education in a proactive, pervasive manner [8].

Constructivist learning scenarios in terms of contextualized learning became technically realizable. They connect abstract learning content to specific use cases and, in particular, to relevant physical situations (situated learning) [9]. This leads to the requirement of adaptive IT solutions to gather, transform, connect, and interpret contextual data as well as react to the learner's current needs in a didactically sensible manner.

2.1 Mobile Adaptive Learning Systems

Adaptive learning systems make use of context information to match the form of presentation, structure, and selection of learning content to a learner's individual preferences, previous knowledge, and learning goals. Two trends of the last years led towards a new generation of adaptive learning systems: Powerful mobile devices and an increased flexibility for the selection and creation of e-learning material.

On the one hand, the access to educational offers by individual, mobile devices with high performance, multiple sensors, as well as a new level of integration into the learner's everyday life allows a broad context acquisition, far beyond the possibilities of previous device classes. Today's smart phones and wearables measure location, orientation, movement, and even physical functions of their users. They have access to calendars, contact data, personal documents, and several other data concerning the organization and social aspects of a learner's everyday life.

On the other hand, these devices create a need for suitable content. Learners are able to access all kinds of information anytime and may choose that piece of information which is most suitable for their current situation. Fortunately, document standards and interoperability concepts like responsive websites allow teachers to create content for different target platforms. Dedicated authoring systems that are partly integrated into learning platforms like, Moodle enable even beginners to make use of these technical possibilities. However, content is hard to present on a device with limited expressive capabilities and to a learner with a limited attention (e.g. during a train ride). Thus, from a didactical perspective, the concept of micro-learning [1] is often preferred in mobile learning arrangements: Small and self-contained learning units that can be worked through easily, even on the move.

The combination of mobile devices and flexible micro-learning content results in adaptivity mechanisms for mobile devices from automated selection of content and context-aware presentation as well as the adaption of learning paths to individual learning styles and goals. Even flexible and adaptive didactic settings for different situations are included [9]. A *mobile adaptive learning system* is an interactive system that personalizes and adjusts e-learning content, pedagogical models, and interactions to meet the individual needs and preferences of users if and when they arise [10], with a special focus on mobile end user devices for context acquisition.

Following the approach of micro-learning, the learning objects and processes become decoupled from the time and place of learning. Yet, the adjustment of educational offers to the given environment, activity, etc. is just a common motivation and benefit of using mobile devices for learning. Detecting the relevant context information (like location, current task or need, surrounding objects and persons, individual learning style and progress, etc.) and analyzing it towards a proper adaptation of a learning offer is a major characteristic of mobile adaptive learning.

2.2 Related Work

A broad spectrum of related work focuses on visions regarding the potential of context-aware systems for education or the utilization of context information for specific learning scenarios [9]. Most of the research in this field can be seen as non-mobile and stand-alone solutions for specific scenarios (e.g. for lecture recordings [11], ITS [12]) or is not suitable for mobile learning due to its complexity [13].

The following related work [14] gives an overview about research in the field of mobile adaptive learning systems. Five pedagogical settings have been identified [15]. The majority of existing mobile context-aware e-learning applications can be classified into:

- *Formalized settings* that take place within the bounds of a dedicated educational institution.
- *Physical settings* that take place in an authentic physical context (e.g. field trips).
- *Immersive artificial* settings where the learning experience is augmented by virtual artefacts, simulations, or pervasive educational games.
- *Collaborative settings* that integrate learners into communities, where collaboration can be supported by recommendation or awareness tools, among others.
- *Loose settings* that take place independent of the current context.

Many existing mobile tools and applications can be assigned to one of the first four settings. They have been developed specifically for selected scenarios, resulting in some rather specialized prototypes, e.g. tangible tabletops [16], personal response systems [17], tools for language learning [18], field trip support [19], museum guides [20], collaborative editing [21], and so forth.

Other research focuses on a more generic approach to educational context-awareness and tries to capture all kinds of context information, including the learner's context, in order to bring forth adaptive frameworks for seamless, personalized learning. For instance, a rule-based system for the selection and presentation of learning content that incorporates a reward/penalize mechanism for further refinement by feedback has been proposed [22]. However, the main focus of this system effectively lies on technical context, such as device specifications, and corresponding adaptation rules for adaptive rendering of learning content.

A similar framework [23] targets an adaptive engine that uses information about learning styles as well as context-awareness in order to provide appropriate, personalized content. What all of these endeavors have in common, though, is that they concentrate on mobile learning in loose settings. So far, no frameworks can be reported yet that encompasses the whole bandwidth of possible didactical settings.

Likewise, the state of research is lacking in analysis of the relation between learning settings and context information in terms of different types, relevance for the setting, availability of the context information, and its reliability. Furthermore, least approaches take into account the teacher's role in authoring e-learning content. However, these considerations are essential to successfully transfer results of e-learning research into current educational development beyond academic visions.

2.3 Roles and Perspectives

Beside technological possibilities as well as limits, specific mobile adaptive learning systems are affected by and the result of an interdependence between authors of learning material (teacher) and mobile users (learner).

Authors of learning material are often employed by educational institutions or by commercial providers of educational material. In particular, teachers come with different levels of IT knowledge. It cannot be assumed that they understand the possibilities and mechanisms of adaptive mobile learning at all. Teaching with adaptive software is a concept that is quite new for teachers and educators nowadays, because of the lack of practical implementations (in the field of mobile adaptive software) that are really used in educational routine [14]. Thus, a suitable adaptive mobile learning system must hide the technical and didactical complexity of adaptation, but at the same time provide tools to connect learning content to contextual information.

Mobile learners are users of adaptive mobile learning systems in terms of consuming and interacting with the learning content, created by authors. They may also be non-technophile and not familiar with mobile adaptation of learning content. While teachers have to be supported in their selection of context and the connection to content, learners may not be aware that their context is used to adjust their learning experience to their current needs. Therefore, the adaptation processes must be invisible to simplify learning on the one hand. On the other hand, unexpected adaptations may irritate learners as they feel they lose control over their devices. This irritation must be avoided by a highly intuitive user interface with a suitable transparency of background adaptation processes. The first step to a user friendly systematic adaptation for teachers and learners is the in-depth analysis of relevant context data for mobile and micro-learning settings.

3 Context

Weiser [24] coined the term *ubiquitous computing* as a concept where computing can occur everywhere and anywhere using any device in any location. From this term the concept of context-awareness was introduced [25]. Context aware devices therefore are trying to make assumptions about the user's current situation and are trying to present services and information based on these assumptions. It is believed that contextual information can be used by mobile learning software to adapt its content to the preferences, needs, knowledge and learning objectives of its user [26].

One must distinguish, however, between context and situation, as context describes just the types of relevant information whereas situation describes the actual values for these types [27]. Detecting the relevant context information (like location, current task or need, surrounding objects and persons, individual learning style and progress, etc.) and analyzing it towards a proper adaptation of a learning offer is a major characteristic of mobile, pervasive learning approaches. Related work [3] compiled an extensive collection of context information that might be relevant for adaptive learning software. Open, however, is the question of what contextual information by teachers are considered to be relevant for creating adaptive learning content. To find out which kinds of information are considered to be relevant by teachers, we decided to analyze different existing educational scenarios.

3.1 Gathering of Educational Scenarios

To have a basis that is as varied as possible and to respect the project's overall goal to develop a framework that may be used in an academically as well as in a business setting, different educational scenarios from both areas needed to be considered. To achieve this, the educational scenarios had to be gathered by a university among teachers and educators as well as adjusted by a business provider of professional learning software. The business provider collected own scenarios among their customers and introduced them into the selection of relevant context information. In order to have these scenarios described in a semi-structured manner, an educational scenario form sheet was developed consisting of:

- an optional graphical representation of the learning scenario itself in form of a Didactic Process Map
- a short description of the scenario's general setting
- a list of desired educational objectives
- a detailed sequence description

In total, we collected 13 distinct scenarios from local teachers of science, humanities, economics and pedagogies as well as 8 scenarios from business and industry. Using these scenarios we evaluated a list of possibly relevant contextual information. An existing classification [27] divides context into the four basic classes of technical, physical, personal and situational context. Because the project has a strong focus on mobile devices, we decided that the mobile part of the physical context had to be considered separately. In a similar fashion, we considered the scenario part of the situational context separately to take into account the scenario-driven analysis approach that we had chosen.

3.2 Identification of Relevant Contextual Information

From the gathered educational scenarios we derived the following subset of context information and sorted them into the aforementioned context classes:

- *Physical context*: weather (current and future), ambient noise, luminosity, time, humidity, temperature
- *Mobile*: location (separated into address, building, region and country), arrival, departure, distance to a point-of-interest, means of transportation, speed, destination
- *Situational context*: body gesture, facial expression, viewing direction
- *Scenario*: learning progress, current task, time required for processing
- *Personal context*: appointments, prior-knowledge (verified and derived), motivation, expectations and motifs, preferences, social relations, disabilities
- *Technical context*: available infrastructure (like printer and external monitor), device capabilities (like voice recording, image capturing, video recording and audio output)

To make the different educational scenarios comparable, we classified them into the five educational settings [15]. Furthermore, we created a questionnaire that was distributed to the contributors of each educational scenario to find out which kinds of contextual information is considered relevant for educational content adaptation by actual teachers, educators and business professionals. The feedback gathered was evaluated and analyzed with respect to relevance. Information from the following context classes has been estimated to be relevant for the defined settings:

- *Formalized Setting*: no significance for any of the context classes.
- *Physical Setting*: Personal, Scenario.
- *Collaborative Setting*: Personal, Scenario.
- *Immersive Setting*: Physical, Personal, Situational, Scenario.
- *Teaching and learning support*: Personal, Scenario.

Additionally, we questioned experts in mobile learning software on how they would estimate the accuracy of detection by today's means, aggregated over the specific educational scenarios. The information from the following context classes was estimated to be accurately measurable:

- *Formalized setting*: Scenario, Technical.
- *Physical setting*: Physical, Mobile, Scenario, Technical.
- *Collaborative setting*: Scenario, Technical.
- *Immersive setting*: Physical, Mobile, Personal, Situational, Scenario, Technical.
- *Teaching and learning setting*: Physical, Scenario, Technical.

Figure 1 provides a graphical representation of the findings from the combination of our data on both relevance and measuring accuracy. The chart is divided into four quadrants of which each one can be described as containing educational settings with context information that is considered to be either:

- *First quadrant*: relevant and accurately measurable.
- *Second quadrant*: relevant and inaccurately measurable.
- *Third quadrant*: irrelevant and inaccurately measurable.
- *A Fourth quadrant*: irrelevant and accurately measurable.

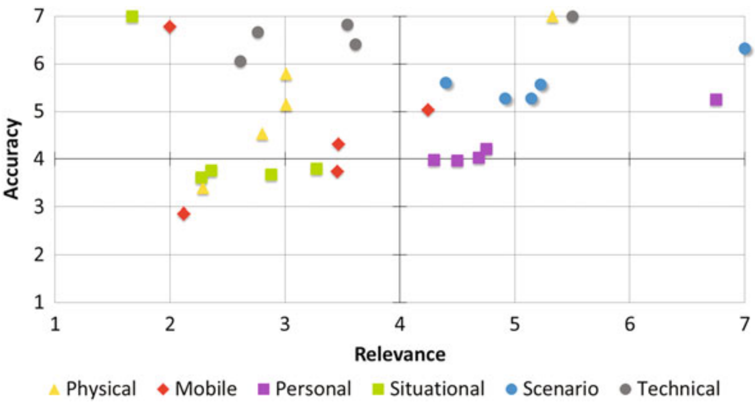

Fig. 1 Combination of the collected data on relevance and measuring accuracy

It is apparent that for further research, context classes that contain information that have been estimated to be either very relevant, very accurately measurable or qualify in both aspects are of utmost interest.

- *Relevant*: Scenario, Personal, Physical (partly), Technical (partly).
- *Accurate*: Scenario, Technical, Physical, Mobile (partly), Personal (partly).

3.3 Implications from the Analysis

Our collected data indicate that the considered information from the scenario context (e.g. learning progress, current task) and the personal context (e.g. prior-knowledge, motivation) have been estimated to be relevant in most of the educational settings. In these settings, information belonging to the scenario context provides an overall good measuring accuracy, and information belonging to the personal context can be measured with at least tolerable inaccuracies.

Most interesting is the fact that information originating from the mobile context (e.g. location, speed, destination), which can be gathered quite accurately with current mobile devices and is often the first choice when it comes to mobile content adaptation, has been estimated to be predominantly irrelevant for individual educational context adaptation. Similar statements can be made for information belonging to the technical and physical context class (except for immersive settings).

This discrepancy can have many reasons. The most obvious could be that this information is indeed irrelevant for educational context adaptation (at least for the moment) and has just been overvalued because it is the most obvious choice when it comes to mobile learning and mobile devices. Another reason could be that teaching with adaptive software is a concept that might be quite hard to grasp for teachers and educators nowadays, because of the lack of practical implementations that have found their way into everyday school life. In the end, research might be facing a so-called “chicken or the egg dilemma” here. On the one hand, it is hard for people to think of features they would like to see in software which is mostly unknown to them or even has an alienating effect. On the other hand, it is difficult to develop useful software concepts when the specific requirements are unclear.

4 Framework

This section deals with the framework proposal for the creation of adaptive mobile learning applications. A closer look will be taken on the general architecture, platform-independent context detection as well as concepts for a dedicated mobile user experience.

4.1 Adaptation of Learning Content

In order to adapt learning content to suit learners’ needs and preferences as well as their prior knowledge and learning objectives, adaptive learning systems make use of several detailed models:

- *Content or domain models* describe what is to be learned within a course or discipline. Causal relations between learning units are modeled as well as distinct levels of difficulty or (methodological) forms of learning.
- *Learner models* keep track of a learner’s progress, that is, their remaining deficiencies with respect to the desired level of knowledge, including mistakes made while solving tasks. Required actions can be derived from these sources of information.
- *Didactic or adaptation models* encompass possible forms of reaction to tackle remaining gaps in knowledge, depending on the given need for action. So far, these reactions are defined only in a rather static form, though.

This schema is inspired by Intelligent Tutorial Systems [12] which combine approaches from cognitive psychology and artificial intelligence. Under the designation adaptive systems, this area of research currently receives an increasing level of attention, factoring in social and pedagogical aspects, in particular. The adaptation of the didactic approach to the learner’s current progress and needs can even be used more effectively by further including information about the learner’s environment and current situation, so-called context. Information about e.g. location, physical parameters, like brightness or noise level, motion, direction, or persons and objects in the learner’s vicinity can be leveraged to a more personalized, while at the same time socially embedded, and thus, more successful learning experience.

A personalized learning experience is created for the learner in several steps. Their needs and preferences are included as well as their current context. Our project's conceptual and technical novelty consists in a multi-staged process, which has not been known from existing approaches and which contributes significantly to the solution's suitability for mobile systems and in particular to its re-usability.

In a first step, an adaptable (i.e. manually adjustable via the variation of certain parameters) application for learning is designed with the help of an authoring system which is to be part of the developed framework. In a second step, this application is deployed onto the learner's mobile device where it serves as an adaptive (i.e. automatically adapting to a given situation) application for learning. Its specific form results from a semi- or fully automatic adaptation to the learner. The core of our solution is the framework for the creation of those learning applications. It will provide a means to generate applications for mobile devices which are then adaptable under certain aspects. This is accomplished on a higher level by an adaptation engine, which again accesses several components:

- *Adaptation rules*: In the relief organization scenario, for instance, a module for vehicle control within a convoy may be shown only to those learners who have a driver's license. This can be modeled by adaptation rules.
- *Learning content and audio-visual material*: To these belong e.g. specific video or audio instructions and explanations for vehicle control in the relief organization scenario.
- *Context detection*: In the above-mentioned scenario, relevant context information might be the specific type of vehicle or the place of action, in order to provide specific content for learning about traffic law in the region.
- *Learning scenario*: Within the learning scenario, the order in which selected learning units may be recommended can be modeled, for instance. Referring to the running example of the relief organization, a general training unit about traffic law could be placed before the more specific training unit about vehicle control. These components can be added or removed in a modular fashion with the help of an authoring system.

4.2 Architecture

Figure 2 depicts a draft of an exemplary implementation of the proposed framework. The system will create a personalized learning experience in a multi-step process and is divided into the three main components authoring system, rule generator and mobile application. The authoring system is used here for teachers to create and manage adaptable (i.e. manually adjustable by variation of parameters) micro-learning content. The created learning contents are transferred to the mobile learning application where they are available as adaptive (i.e. automatically adjustable) learning content.

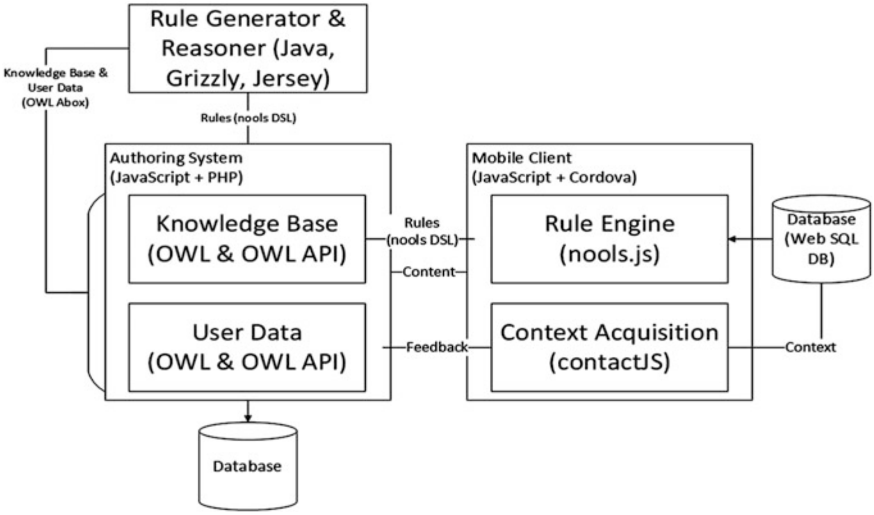

Fig. 2 The architecture of the framework with its server side components on the *left side* and the client side components on the *right side*

In order to implement the adaptivity of the mobile learning application, conclusions must be drawn both from the user behavior as well as the collected context information. For this purpose, a reasoning mechanism is necessary which, however, cannot run on the mobile device itself due to the limited resources of mobile devices and the lack of a platform-independent reasoning framework. Outsourcing the reasoning on a server to only trigger it when necessary is not possible either because it has been identified as an essential requirement of the application to be usable without an internet connection.

To meet these requirements, a hybrid approach has been chosen. Thus, the reasoning is indeed outsourced to a server, but is triggered by the rule generator just before deployment to the mobile device. This uses both the knowledge base and possibly collected feedback about the behavior of the learner and creates a set of rules. The rule set is evaluated by a rules engine in the mobile learning application to dynamically select or adjust learning contents. The inference of high-level contextual information takes place on the mobile device as part of the context acquisition. Figure 3 shows a simplified representation of the data flow.

A variety of programming languages, frameworks and technologies will be used hereby:

- *Java* will be mainly used for the rule generator. In combination with frameworks like Project Grizzly and Jersey, it has been used to build a RESTful web service that provides interfaces for rule generation and information about different parts of the adaptation engine or context detection (e.g. supported contextual information).

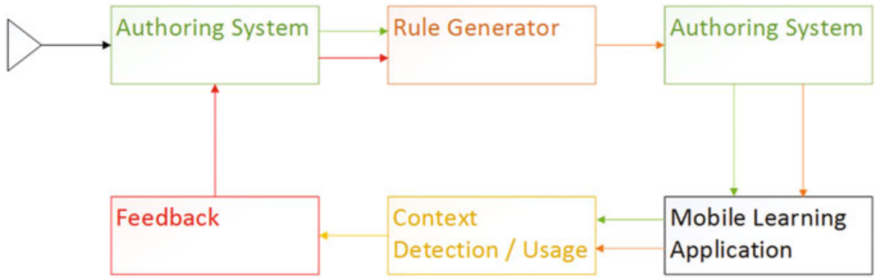

Fig. 3 The data flow of the complete infrastructure. Rules at first are solely generated from the content created by an author. In a later step, collected information about the user's behavior (feedback) can be used to refine the originally generated rules

- *JavaScript* is used in many different parts of the whole project. On the server side, the authoring system leverages JavaScript to create the user experience. On the mobile client side, JavaScript is used for the context detection (see Sect. 4.2).
- *OWL* the Web Ontology Language is a specification of the World Wide Web Consortium to create, publish and distribute ontologies using a formal description language. It is based on RDFS and part of the W3C Semantic Web Activity. OWL is used in combination with the OWL API to store the knowledge base and the collected user behavior as well as to draw conclusions from these data (reasoning) for adaptivity.
- *SPARQL* is a query language for RDF. It is used, among others, to combine information about the user's behavior and the collected contextual information.
- *Nools* is a rules engine based on the Rete algorithm written entirely in JavaScript. It is used to evaluate the rules that were created by the rule generator and accordingly give proposals for adaptations.
- *Apache Cordova* is a set of device APIs that allow a mobile app developer to access native device function such as the camera or accelerometer from JavaScript. It is used in combination with the context detection and rule engine to create the adaptive mobile learning application infrastructure.[1]

4.3 ContactJS[2]

Modern mobile devices like smart phones or tablet computers, that are already being used to consume mobile learning content, are equipped with a variety of

[1]https://cordova.apache.org/.

[2]The work in this part of the project was implemented as part of a master's thesis. Stefanie Lemcke, who worked on the master's thesis, has been supported in her work by the *Stiftung Industrieforschung* with a scholarship.

sensors and interaction options that can be used to gather contextual information. When it comes to the detection of such information, a challenge, however, is to address the heterogeneous landscape of devices and operating systems (and thus programming languages) that learners are using.

Currently, there are many different smart phone and tablet computer developers in the market each using their own operating systems (e.g. iOS,[3] Android,[4] and Windows Phone[5]) and programming languages (e.g. Objective-C, Java, and C#). A context detection system should support all or at least the most commonly used platforms. Development and maintenance of such a system individually for each programming language would be an expansive task and almost impossible to afford by individual educational institutions.

Fortunately, modern mobile devices are supporting the concept of so-called web applications. Such applications are using technologies like HTML, CSS and JavaScript normally used for the development of web sites. In combination with frameworks like Cordova it is possible to create mobile applications that can be ported to different operating systems at reasonable expense.

There is a range of existing frameworks that can be used for context detection like Context Toolkit [28], Citron [29], CASanDRA [30], LoCCAM [31], CASS [32], Hydrogen [33], Gaia [34], Context Fabric [35], CoBrA [36], Solar [37], SOCAM [38] and JCAF [39], to name a few. Most of these frameworks are based on a client-server model with distributed context detection. Solely LoCCAM offers an infrastructure for context detection on mobile devices but was only developed to be used on devices running the Android operating system. Moreover, only Context Toolkit, Context Fabric, CoBrA, LoCCAM and JCAF had usable source code available so that the remaining frameworks where excluded from further consideration.

Due to our focus on context detection and the fact that CoBrA and Context Fabric are also offering inseparable reasoning mechanisms, that exceed the targeted functionality and are too complex to port, we decided to focus on Context Toolkit, LoCCAM and JCAF as potential candidates for a port.

- *Loosely Coupled Context Acquisition Middleware* (*LoCCAM*) is an infrastructure for context acquisition for devices that are running the Android operating system. Its main focus is to use smart phones as the main element for context detection. This may include local and remote resources as sources of information. The systems context detecting components are self-regulating which is achieved through the use of the OSGi framework as the runtime environment. The detection and processing of actual contextual information is realized through Context Acquisition Components (CAC). The collected contextual information is managed and published with the use of System Support for Ubiquity (SysSU[6]).

[3]http://www.apple.com/ios/.

[4]http://www.android.com/.

[5]https://www.windowsphone.com/.

[6]https://code.google.com/syssu/.

- *Java Context Awareness Framework* (*JCAF*) is a Java-based framework for context detection and publishing. It is divided into two parts: the Context-awareness Runtime Infrastructure and the Context-awareness Programming Framework. The first part provides the developer with a basic frame for developing a context-aware infrastructure. It is designed as a service-oriented architecture and communication is realized through Java RMI. The second part is an API that provides interfaces for context-aware applications to communicate with the framework. Both parts of the framework are together implemented as three-layer architecture. The Content Client Layer acts as the API, the Content Service Layer manages entities as well as context transformation and aggregation, and the Context Sensor and Actuator Layer task is to actually gather contextual information or to manipulate the context.
- *Context Toolkit* is a framework developed for the use of distributed context detection hardware [28] and can be seen as one of the first frameworks developed for that purpose. Its main focus is on context detection and processing. The framework primarily consists of the five components Widget, Service, Aggregator, Interpreter and Discoverer. Widgets and Interpreter thereby have the tasks of context detection and processing, Services are used to execute orders and actions, Aggregators are used to combine different data for a single entity and Interpreters are used to format and interpret the collected data. The components are realized as abstract Java classes that provide all the basic functionality where concrete features need to be implemented by the developer.

Before we could select a framework to be ported some functional and non-functional requirements for the resulting system had to be noted. Functional requirements are:

- *Context detection*: The system needs to be able to detect contextual information that can be used by learning software to adjust its content to the preferences, needs, knowledge and learning objectives of its user. Sources for that information should be sensors as well as software components.
- *Context persistence*: There is a need for a structure to save gathered contextual information either for a short time or long periods of time.
- *Context processing*: Initial processing steps like interpretation or transformation of the gathered data should be possible. This should be limited to simple conversions like changing the format of the data or for example retrieving the address for geographic coordinates. Complex abstraction work will be done by the hosting application.
- *Context publishing*: For context-aware applications to react to gathered contextual information, it is important to be able to query certain information. This should be possible on demand or if needed in a callback-like manner.
- *Communication*: The interaction between the components of the framework needs to be adapted for the use on mobile devices. It is important to be able to interact with existing hardware and software components of the device.
- *User context management* (*optional*): To protect the user's privacy, it should be possible to inspect and manage the gathered contextual information. This means

that the user would be able to delete information and to add or remove certain context sources. Because context aware applications rely heavily on contextual information it could be problematic if a user refused to reveal required information. This deliberation took us to the conclusion that the decision of which information to gather and which not is in responsibility of the hosting application and that such management on the detection layer can be seen as optional.

The non-functional requirements are:

- *Platform independence*: A main goal of this project is to create a context detection system that can be used on a wide variety of platforms.
- *Operability and maintenance*: The resulting system is intended to be a basic framework for context detection. It will provide an environment to detect and publish contextual information where concrete implementations like the connection of a sensor need to be done by the user.
- *Flexibility*: Supporting a wide range of context sources from both software and hardware is desirable. This includes a simple but open structure for data persistence.

Table 1 shows an overview of the most important features for each of the frameworks. Each framework focuses on the detection and publishing of context information. Thus, independence between the collection of context data and the usage of such data by a context-aware application is given. Despite this similarity each of the frameworks has its own strong points. LoCCAM already focuses on mobile devices (Android only, though) and uses a self-regulation mechanism to start and stop context detection components. JCAF uses a service oriented infrastructure for context detection and publishing and focuses on distributed systems. Context Toolkit uses a concept similar to JCAF but provides separate modules for context processing and registration and is free of any dependencies. Ultimately, the decision was made in favor of Context Toolkit. Mostly because it is the only framework that had no dependencies that would have been needed to port as well.

As previously stated, our goal was to port an existing context detection framework to JavaScript. A strong focus was set on platform independence to support as many mobile devices as possible. However, two questions needed to be answered first before a port could be done.

Table 1 Prominent features of the port candidates

Feature	Framework		
	LoCCAM	JCAF	Context toolkit
Language	Java	Java	Java
Application	Mobile devices (android)	Distributed systems	Distributed systems
Modules	CAC (detection and processing)	Monitor (detection) services (processing) actuators	Widget (detection) service (actuator) aggregator interpreter
Dependencies	OSGi, SysSU	Java RMI	

The first question that arose was which implementation variant to choose. On the one hand, the framework could have been implemented as a standalone application which would run on the target device. The advantage would be that only one context detection system would be running on the device providing contextual information to all context aware applications. A serious disadvantage would be that each mobile operating system has another way of distributing data between running applications. A context detection application would need to take each of these into account and thus would limit its platform independence.

On the other hand, an implementation as a library was conceivable. The library would be included by future context-aware applications and provide all the functionality needed to gather contextual information. Such a solution could be designed to work around the problem of distributing the gathered contextual information between individual applications because each application that used the library would have direct access to the gathered data. The main disadvantage of such an implementation would be that different applications using the library would gather and store their own data and so would inevitably produce redundant data. Eventually, the decision fell in favor of to the library solution because the number of concurrent e-learning applications can be expected to be low and it would bypass the data distribution problem, yet could be used to implement a central context detection application later.

The second question that had to be answered concerned the components to be adopted. Basically, we planned to take over the existing structures and adapt them only to the selected technology. It turned out, however, that some components and concepts conceived to realize the communication in a distributed system would be unnecessary or needlessly complicated in a system where all parts are situated on the same device. With that in mind, the following components were selected for porting:

- *Widgets* are the main components of the Context Toolkit. Their task is to gather contextual information and they are therefore directly connected to context sources such as sensors. They were fully ported, including the structures for the modeling of the detected data and the control of the communication between the components.
- *Aggregators* are consolidating context information about a single entity like a person, a location or an adaptation rule. Hence, they are context gathering components, as well. Contrary to Widgets, they are not directly connected to a context source but rather are collecting different information that is provided by Widgets and Interpreters. Aggregators were ported closely to the original system. Again, the main task will be the consolidation of contextual information. Furthermore, Aggregators will manage the communication with external applications that are using the framework as well as the persistence of the gathered information.
- *Interpreters* process information to achieve format conversions or to abstract low-level context into high-level context. They can be accessed by Widgets or

Aggregators and the complexity of the abstraction process is to be chosen by the programmer.
- *The Discoverer* acts as a registry and provides an overview of all available Widgets, Aggregators and Interpreters. The Context Toolkit allows the existence of multiple discoverers. This was necessary because in a distributed system, parts of the network could be safeguarded, which might prevent communication with a global discovering instance. However, on a mobile device where a discovering service would be accessible to every other component, such a problem would not arise and thus the system can rely on a single Discoverer. In addition to the features defined in the original framework, the ported version covers the communication between different components.

The following components were omitted:

- *The BaseObject* manages the details of the communication in a distributed system. These details include the underlying transport protocol and the exchange format used to transport data through the network. In a local system, all components communicate directly and thus the use of the BaseObject component is obsolete.
- *The DataObject* encapsulates every bit of data that is sent between the components of the system into a general data structure and hereby prepares the data to be transformed into XML and accepts the data retrieved from parsed XML, respectively. This covers requests to as well as responses returning from a component. Again, in a local system such a transformation is not necessary, because all components communicate directly.
- *Services* in Context Toolkit act as actuators and are used to change or influence the environment. As this is not a requirement of the project, the component was not ported but could possibly be added later.

For the implementation of the proposed JavaScript library, a few existing frameworks were used. These are:

- *RequireJS* was used as a file and module loader for JavaScript. Its main benefit is that JavaScript files can be easily managed in a modular way similar to Java-Imports. Additionally, required JavaScript files are loaded on demand. This results in cleaner code and improved processing speed.[7]
- *ease.js* brings object-oriented concepts like classical inheritance, abstract classes and methods, interfaces and static and constant members to JavaScript. It was used to ease the port from the class-based programming language Java to the classless script language JavaScript.[8]
- *r.js* is an optimizing tool as a part of RequireJS. It can be used to minimize, optimize and combine different modules into a single JavaScript file. It was used to generate the final single library file.

[7]http://requirejs.org/.

[8]https://www.gnu.org/software/easejs/.

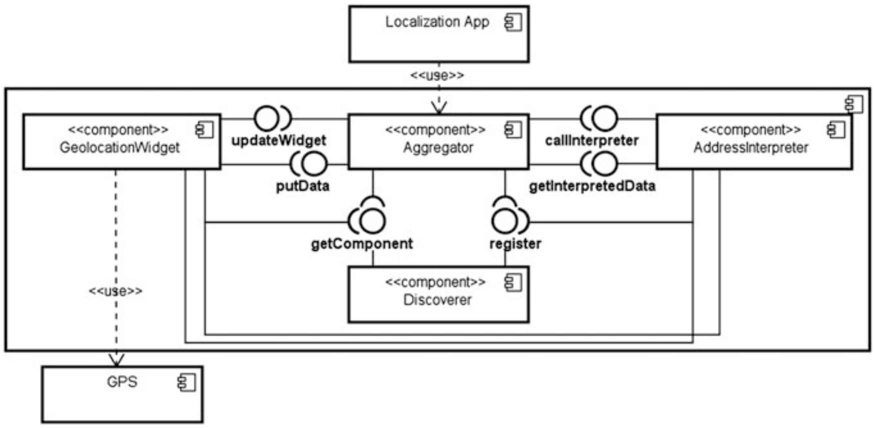

Fig. 4 An example configuration of an application using contactJS to retrieve the address of the user's current location

- *Math.uuid*[9] offers functionality to generate Universal Unique Identifiers (UUID) in JavaScript. UUIDs are used in Widgets, Aggregators and Interpreters and are required for the registration and search by the Discoverer. The library supports the creation of UUIDs following the RFC 4122 Version 4 [40].

The implementation follows the concept that was explained above. The most important changes, compared to the original system, have been made in the area of communication. The components solely required for the communication in distributed systems have been discarded and the concept has been adapted to the communication on local systems.

The Discoverer is now a central component of the framework. Every other component has to register itself with the Discoverer, and future communication is established by acquiring required all components from the discoverer through the use of UUIDs. Widgets and Aggregators have also undergone some changes. Widgets don not save a history of past contextual information anymore; this is now done by the Aggregator. Aggregators are now also responsible for the communication with applications that are using the framework.

Figure 4 shows an example configuration of an application using contactJS. In this example, the application is using the framework to acquire the address of the current position of the user, which might be relevant on a field trip. A widget was deployed to acquire the position of the user directly from the GPS sensor of the mobile device and an interpreter encapsulates the reverse geocoding functionality provided by an external source (e.g. Google or OpenStreetMap).

[9]http://www.broofa.com/Tools/Math.uuid.js.

Both components are registered with the discoverer and are publishing their information to the aggregator which in turn communicates with the hosting application. Similarly, other types of context can be managed.

4.4 Mobile User Experience

As mentioned earlier, adaptive mobile learning applications are able to detect the user's situation by evaluating contextual information and using this information to select and adjusting content accordingly. Typically, this will happen in the background unnoticed by the user.

In Sect. 3, we have already shown that the information used for the adaptation processes is often limited in terms of accuracy and reliability. This can lead to the application behaving in a way that is no longer understandable or even alienating for the user. Moreover, the detection of contextual information can pose the risk of violating the user's privacy.

To prevent these problems, the collected contextual information and the adjustments provided on the basis of this information must be transparent for the user. The challenge hereby is to adjust the flow of information in such a way that the user will not get overloaded with information, and thereby deflected from the actual learning task, yet will not be under-served with information, which would create the aforementioned problems.

In the course of the project, we tried to identify user experience design patterns that tackle the problem of information supply and privacy violation caused by a possible lack of transparency. At the beginning of the development of the framework's user experience component, the functionality to be addressed was being determined from nominal patterns of adaptive user experience. Those patterns were elaborated from various user-approved examples.

A digital paper-prototype of an example scenario for adaptive mobile applications was created and made a representative study on the implemented principles possible. It features the following three key techniques:

- *Common user interface components*: Consist of components chosen from the platform's default UI set. Conforming to the platform's interface guidelines, these components are widely proven.
- *Real-life context simulation*: Simulated adaptations based on context changes that happen to the most mobile users every day.
- *Example scenario*: Features a real content scenario that is already adopted in existing applications.

Hereinafter, the user experience patterns that were used or developed, shall be explained in more detail.

On-boarding. As seen in many current mobile applications, an on-boarding process is used to make users familiar with interaction patterns that they would not have been exposed to on a regular basis. The on-boarding is often a set of different

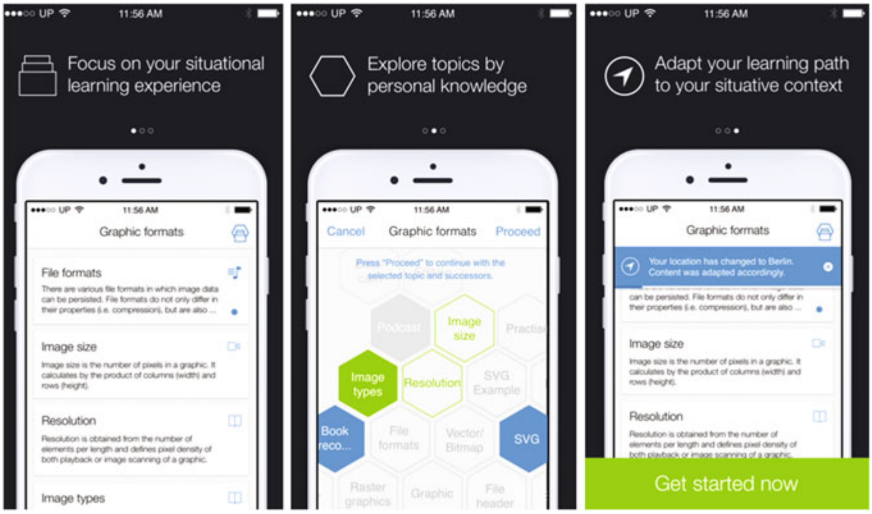

Fig. 5 The on-boarding views that are highlighting our focus on situational learning, the explorative honeycomb navigation view and the way newly detected contextual information is displayed

informational screens shown to the user on the first application start. They provide a brief introduction to the application and highlight and explain the key features. Again, the goal of the on-boarding screens is to make the user familiar with the things he does not see in most of the applications he is already using. In this case we decided to highlight our focus on situational learning, the explorative honeycomb navigation view (see Fig. 5), and the way newly detected contextual information is displayed.

Tutorial overlay. Another user experience pattern that is often used to introduce users to certain app features are so-called coach marks. They act as an indicator for the actions that can be performed on the application and explain what the controls on the application are meant for. We decided to use these coach marks in form of a tutorial overlay screen, a black half transparent screen that lies on top of the actual controls and is presented at the first application start.

Learning content navigation. Former research [41] identified that learning and teaching styles differ in the way learning content is presented to and grasped by the learner. A distinction is made among others between a sequential and global learning and teaching style. Sequential learning hereby "[…] involves the presentation of material in a logically ordered progression, […] When a body of material has been covered the students are tested on their mastery and then move to the next stage." To support this classical way of learning we developed the *Learning card stream* (see Fig. 6).

Every card in the stream represents a micro learning unit. New cards are added on top of the stream when a card is consumed or contextual information changes.

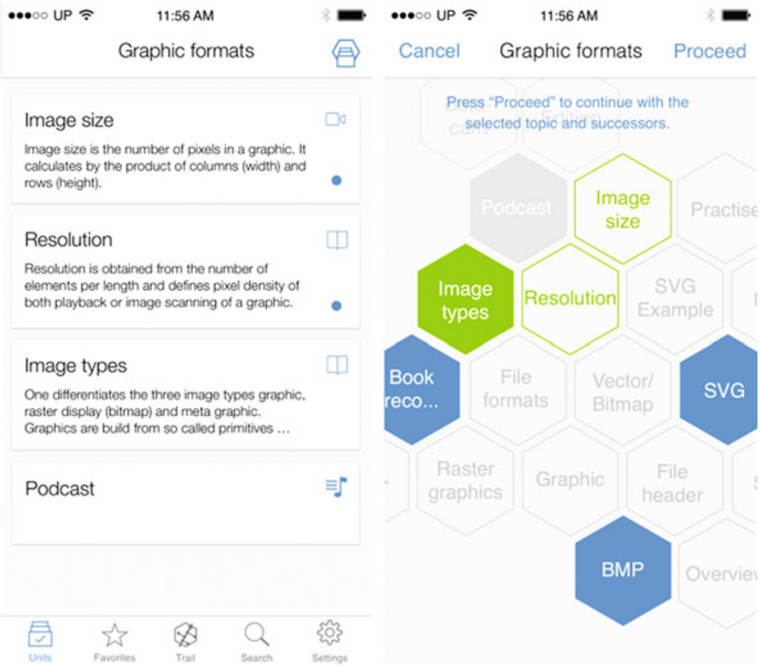

Fig. 6 On the *left* is the card stream to support sequential learning and on the *right* is the honeycomb view that aims at supporting global learners

New cards are flagged to be easily recognizable. To give the learner a general idea of what content might await him, every card has a title, a short description and an icon that will hint at the content type. Though a linear learning style is in focus here, a degree of freedom is given as older cards can be explored through backtracking.

In contrast to a sequential learning style where learners "[…] follow linear reasoning processes when solving problems […] learn best when material is presented in a steady progression of complexity and difficulty" [41] learners who follow a more global style of learning tend to "[…] learn in fits and starts […] make intuitive leaps […] sometimes do better by jumping directly to more complex and difficult material." [41]. To support such a global learning style, we developed the honeycomb navigation view.

All learning units of a scenario are arranged as a honeycomb structure where related learning units are positioned in immediate vicinity. This enables the learner to take a glance at future learning units. Currently reachable learning units are colored blue. Selecting such a learning unit will display direct follow-up learning units, another way to estimate the following educational trail. Which learning units will follow is based on contextual information and the relations between learning units that were defined by the author. We hope that this will help the learner to adjust the learning content to their own needs and to find individual learning paths.

As it is hard to predict which learning style the user is favoring at the moment, we decided to make it possible to switch between the two navigation schemes at any time.

Alternatives. In an adaptive learning scenario, it is likely that alternative versions of the same learning content are available. The availability of these versions is based on the content (e.g. used media types) or contextual information. Although the application may initially make assumptions about which version would be most suitable for the learner, we came to the conclusion that it is important for the learner to freely switch between alternatives if he wishes to. The list of alternatives (if available) can be accessed by tapping on the content type icon of a card. For each alternative, the contextual information which is the condition for the learning unit as well as the content type is shown. Moreover, a thumbnail is shown to help the learner identify content that is most suitable for him.

Calibration. It can be necessary to pre-asses the learner to identify his prior knowledge. Unfortunately in a mobile learning scenario this is hard to achieve by questionnaire. On the one hand it is hard to create a generic questionnaire that fits all possible scenarios and on the other hand users usually aren't willing to endure a long question session just to consume rather short learning content.

To tackle this problem we decided to use the existing learning units as something similar to a questionnaire. After the on-boarding the learning units are listed to the user. As we are following the concept of micro-learning each learning unit will treat a small and manageable learning content. The user may look at a preview of the content of each unit and check off the units he already knows. This is by no means a perfect solution but rather provides as a basis for further research.

Another need for calibration arises from the fact, that contextual information may be subjective, e.g. ambient noise is perceived as disturbing by different users to varying degrees. The combination of gathered contextual information and user behavior already provides some hinds that can be used to automatically and manually adjust learning content to be selected in situations that are more suitable for learning.

In addition we decided to use the gathered contextual information for a given situation and occasionally ask if that situation is hindering concentration of the user (see Fig. 7). This provides a direct feedback by the user on which situations are ideal for learning and which are not.

Context change notifications. As was stated at the beginning of this section, it is important to inform the learner about newly gathered or updated contextual information. Above all, this feature aims at creating awareness for the contextual information that is gathered by the application. This, in return, enables the learner to identify privacy violations but also erroneous information.

In order not to distract the user from their actual learning task, the notifications were designed to be as non-obstructive as possible while maintaining a decent amount of information. As good as our intentions with these notifications are, we

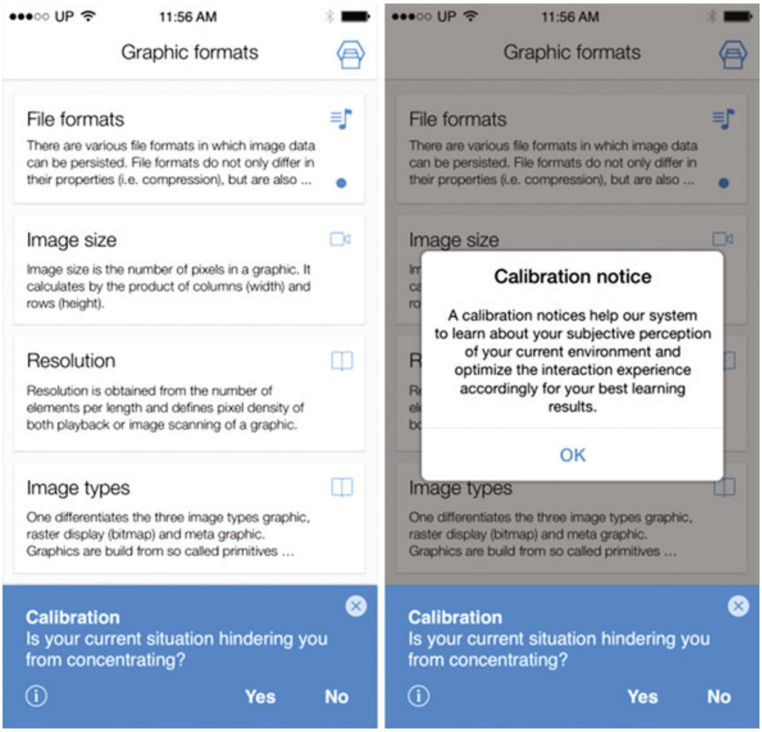

Fig. 7 On the *left* is the calibration prompt to gather direct feedback by the user on which situations are ideal for learning and on the *right* is a notification informing the user about the calibration

cannot assume that every user would like to be notified. Thus context change notifications will disappear automatically after a short while and will also present the user an option to disable notifications altogether.

What yet need to be evaluated is what kind of context information is suitable for these notifications and which notification frequency is acceptable for the user. It can be safely assumed that in most learning scenarios every second notification about changing ambient noise levels would soon be perceived as annoying (see Fig. 8).

Tracking events. As important as it is to make the user aware of changing contextual information, it is equally important to provide an easy to understand history of events and contextual information gathered in the past. Such a history function is represented by our *Trail* feature. The *Trail* chronologically keeps track of all past events and the contextual information that was gathered. In addition to an overview of the gathered data, the user can select single events or pieces of information and revise assumptions made by the context detection (e.g. delete information, mark information as false or change detection settings) (see Fig. 9).

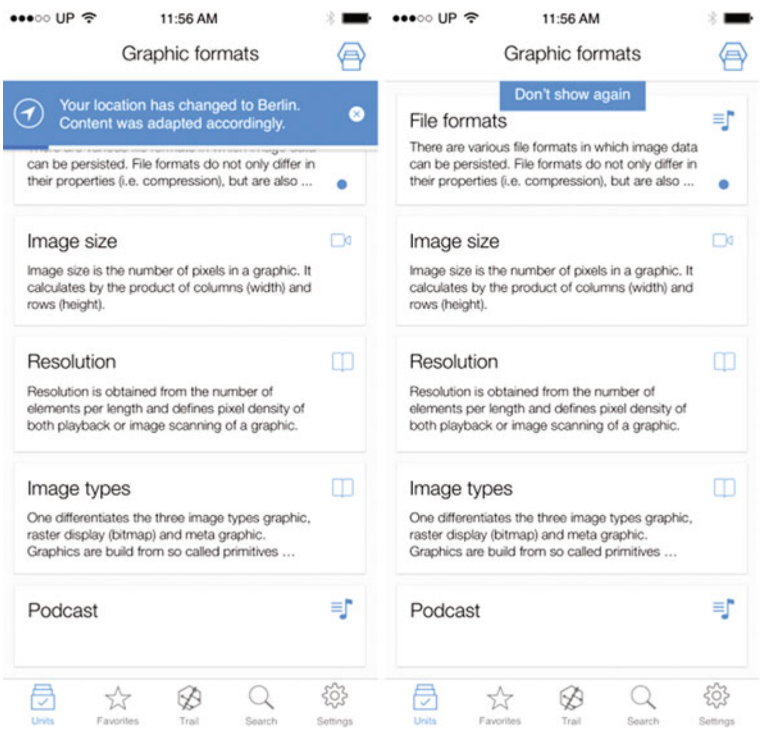

Fig. 8 An in-app notification informing the user that his current location has changed

5 Future Work

As the project is still in progress, some aspects will need some further work or refinement. The major open ends concerning context detection accuracy, cross-platform context detection, authoring user experience and mobile user experience will be addressed in this section.

5.1 Context Detection Accuracy

Some estimation about context detection accuracy that were made at the beginning of the project need to be reconsidered. That is no surprise since technology is in continuous change, and therefore makes ever more sources of contextual information available and may increase the accuracy of already detectable information, respectively.

For example, in the field of facial, emotions and gesture detection many new devices and technologies, like Oculus Rift, Myo, Leap Motion and Windows Hallo,

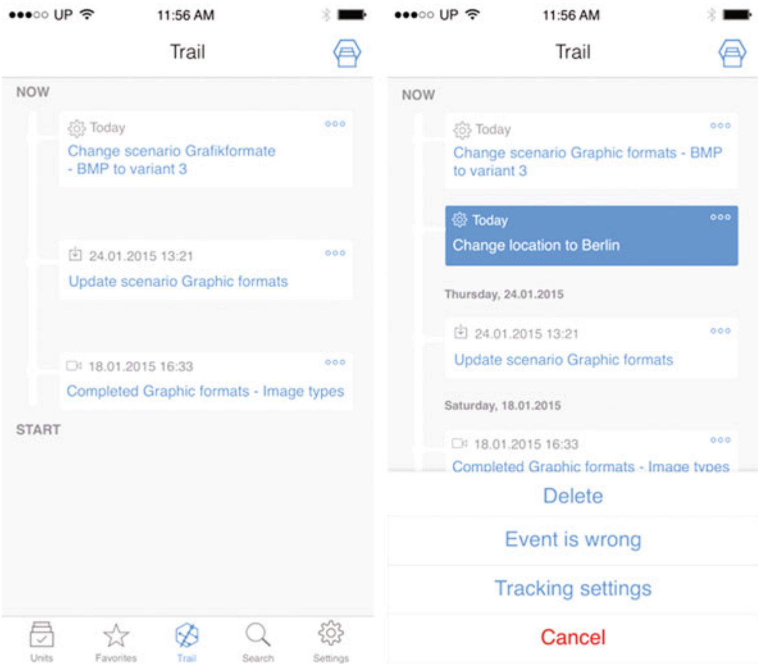

Fig. 9 The *Trail* feature chronologically keeps track of all past events and additionally lets the user revise assumptions made by the context detection

surfaced. These devices and technologies have made a lot of information available that had, until then, not been measurable at all. The trend towards wearable computing even promises many more sensors that will be available for use in the future.

Fortunately, our context detection was designed from the ground so that new contextual information can be added at any time. Nevertheless identifying new contextual information and incorporating software components to detect this information will be an ongoing task for developers that will be using the framework.

5.2 *Cross-Platform Context Detection*

Our context detection framework *contactJS* provides all means that are necessary to easily deploy a platform independent infrastructure to gather contextual information which can be used by mobile context-aware learning software to adapt its content to the preferences, knowledge and goals of its users.

Though contactJS in its present form is already a capable framework for the development of context-aware applications on mobile devices, further improvements are possible. This applies mainly to mechanisms that make the use of contactJS more convenient, such as the automatic detection and configuration of widgets and interpreters or the possibility of synonyms for certain contextual information. The latter will make it much easier to use widgets or interpreters that were created by developers who designed certain pieces of contextual information in a different way.

It is planned to release the context detection component (as well as the rest of the framework) as open source so that it is possible for the software development community to implement own features or context detection components.

5.3 Authoring User Experience

Developing a pleasant user experience for the authoring system is important because especially non tech-savvy users should be able to use the system. We are currently developing concepts and prototypes to evaluate a system that makes it possible to design mobile adaptive learning scenarios without the need to be firm with things like ontology reasoning or Boolean logic.

The main challenge is to map the abstract concepts of adaptivity and context detection, but also the relations between the various micro-learning units into an intuitive user interface. Furthermore design patterns need to be developed to support teaching staff to create effective micro learning content.

5.4 Mobile User Experience

The evaluation of the mobile user experience patterns that we developed is well underway. Initial feedback has already been positive, but also revealed some shortcomings. More tests are needed and are currently performed.

It has been found out that some of the design assumptions (e.g. using standard user interface components, or the support of irregular learning behavior) have proven to be right, it also became apparent that certain concepts (e.g. “Honeycomb” Navigation or onboarding) not completely satisfy the anticipated effects yet.

Regarding the core requirement of improving the transparency of context acquisition and adaptation, no definitive statement can be made. This requires more extensive testing with “real” students and “real” learning content. From the interviews with experts, however, it appears that context acquisition and adaptation are noticed neither positive nor negative. This can already lead to the conclusion that in order to achieve the goal of raising awareness of these mechanisms, further adjustments to the concepts will be needed. This could be made possible for example by an advanced primary onboarding with a clearer focus on adaptivity and context detection, as well as a secondary onboarding on learning scenario level to introduce in advanced methodology adaptation and scenario structure. Also the use of embedded help structures (e.g. coach marks) for interactive learning of application-specific navigation structures and to locate the different functions is to be classified as effective.

As our current patterns were developed on the basis of the iOS user interface, patterns need to be developed to transfer native GUI elements to other platforms (e.g. iOS tab bar to Android pop up menu).

6 Conclusion

Context-aware systems become more and more relevant for educational purposes. In particular, the trends towards mobile assistants with their broad spectrum of sensors and information retrieval techniques open broad possibilities for context acquisition and application on mobile devices.

The research visions of context aware systems for education or the utilization of context information for specific learning scenarios are impressive and innovative [9]. But, a transfer into actual, real world learning scenarios is often limited to dedicated educational settings.

This chapter was dedicated to the systematic realization of *mobile adaptive learning systems*. These are interactive systems that personalize and adjust e-learning content, pedagogical models, and interactions to the user's individual needs and preferences with a special focus on mobile end user devices for context acquisition. The perspectives of learners, teachers and developers on mobile adaptive learning systems have been discussed and evaluated—with a focus on the context data itself and user interfaces for learners and teachers.

A systematic approach for developing mobile adaptive learning applications has been introduced. In a first step, relevant contextual information has been identified by analyzing several educational scenarios from both, a technical point of view and a pedagogical perspective. In a second step, this context information has been classified regarding the current technical possibilities to acquire and utilize it an accurate manner. The findings of both steps led to a framework that supports even non-technophile authors or teachers in their creation of context-aware learning applications for mobile devices. Both, learner and teacher, benefit from highly intuitive interfaces that allow the utilization of such very technical concepts like context acquisition without requiring any deeper knowledge thereof.

However, the work on the framework continues, as current and future trends in e-learning offer potential for expansion and application. One of these trends, which represent a promising application for adaptive micro learning applications, is *Industry 4.0* (for example in the form of the resilient factory, intelligent maintenance management, networked production, etc. [42]). These scenarios typically provide an early and persistent need for training directly in the work process in order to limit costs and expenses to a manageable minimum. Adaptive mobile adaptive learning applications inherit the potential to satisfy these requirements.

Another promising current trend is the use of proactivity in software solutions. This means that an application is enabled to predict the behavior of the user and herby can offer information or answer requests before they have been explicitly stated. Prominent examples of this are so-called digital assistants (e.g. Google Now,[10] Cortana,[11] and Siri[12]). In the context of the framework, the combination of the collected context information and the actions of the learners could be used to predict future learning behavior, which in turn would allow for a more precise adaptation of the teaching content.

Finally, a recurring trend is competence-based assessment. Since competencies are commonly mapped in the form of ontologies, this may represent an elegant way to map prior knowledge as well as learnable skills into the form of contextual information. Current research deals with the development of cross-platform competence-based assessment [43].

Acknowledgments We would like to thank Helena Jank, Stefanie Lemcke, Martin Biermann and Julius Höfler for the invaluable work that they have contributed to the project. We would also like to thank the BMWi for the funding of the MOTIVATE project[13] and the Stiftung Industrieforschung for supporting Stefanie Lemcke's master thesis with a scholarship.

References

1. Hug, T.: Didactics of microlearning. Waxmann Verlag (2007)
2. Dey, A.: Understanding and using context. Pervasive Mobile Comput. **5**(1), 4–7 (2001)
3. Economides, A.: Adaptive context-aware pervasive and ubiquitous learning. Technol. Enhanced Learn. **1**(3), 169–192 (2009)
4. Ralston, A.: Interactive learning strategies for mobile learning (2013). http://www.unesco.org/new/en/unesco/themes/icts/m4ed/unesco-mobile-learning-week/speakers/anthony-ralston/
5. Shute, V.J., Psotka, J.: Intelligent tutoring systems: past, present, and future. Technical report, DTIC Document (1994)
6. Mitrovic, A., Koedinger, K.R., Martin, B.: A comparative analysis of cognitive tutoring and constraint-based modeling. In: User Modeling 2003, pp. 313–322. Springer, Berlin (2003)
7. Holt, P., Dubs, S., Jones, M., Greer, J.: The state of student modelling. In: Student Modelling: The Key to Individualized Knowledge-Based Instruction, pp. 3–35. Springer, Berlin (1994)
8. Lucke, U.: Pervasive learning. In: Pervasive Adaptation Research Agenda for Future and Emerging Technologies. Th. Sc. Community (2011)
9. Lucke, U., Specht, M.: Mobilität Adaptivität und Kontextbewusstsein im E-Learning. i-com **11**, 26–29 (2012)
10. Stoyanov, S., Kirchner, P.: Expert concept mapping method for defining the characteristics of adaptive e-learning: Alfanet project case. Edu. Tech. Res. Dev. **52**(2), 41–54 (2004)
11. Aguilera, N.E., Fernandez, G., Fitz-Gerald, G.: Addressing different cognitive levels for online learning. In: ASCILITE, pp. 39–46 (2002)
12. Pinkwart, N., Aleven, V., Ashley, K., Lynch, C.: Adaptive Rückmeldungen in intelligenten Tutorensystem largo. eleed 5(1) (2008)

---

[10] http://www.google.com/landing/now/.

[11] http://www.microsoft.com/en-us/mobile/campaign-cortana/.

[12] https://www.apple.com/ios/siri/.

[13] Funding reference: KF3155601MS3.

13. Aroyo, L., Dolog, P., Houben, G.J., Kravcik, M., Naeve, A., Nilsson, M., Wild, F.: Interoperability in personalized adaptive learning. J. Edu. Technol. Soc. **9**(2), 4–18 (2006)
14. Moebert, T., Jank, H., Zender, R., Lucke, U.: A generalized approach for context-aware adaption in mobile e-learning settings. In: 2014 IEEE 14th International Conference on Advanced Learning Technologies (ICALT), pp. 143–145. IEEE (2014)
15. Lucke, U., Rensing, C.: A survey on pervasive education. Pervasive Mobile Comput. **14**, 3–16 (2014)
16. Dillenbourg, P., Evans, M.: Interactive tabletops in education. Int. J. Comput. Support. Collab. Learn. **6**(4), 491–514 (2011)
17. Martyn, M.: Clickers in the classroom: an active learning approach. Educause Q. **30**(2), 71 (2007)
18. Uther, M., Zipitria, I., Uther, J., Singh, P.: Mobile adaptive call (mac): a case-study in developing a mobile learning application for speech/audio language training. In: IEEE International Workshop on Wireless and Mobile Technologies in Education, WMTE 2005, p. 5. IEEE (2005)
19. Giemza, A., Bollen, L., Seydel, P., Overhagen, A., Hoppe, H.U.: Lemonade: a flexible authoring tool for integrated mobile learning scenarios. In: 2010 6th IEEE International Conference on Wireless, Mobile and Ubiquitous Technologies in Education (WMUTE), pp. 73–80. IEEE (2010)
20. Oppermann, R., Specht, M.: Adaptive mobile museum guide for information and learning on demand. In: HCI (2), pp. 642–646 (1999)
21. Steimle, J., Brdiczka, O., Mühlhäuser, M.: Coscribe: integrating paper and digital documents for collaborative knowledge work. IEEE Trans. Learn. Technol. **2**(3), 174–188 (2009)
22. Zhao, X., Jin, Q., Okamoto, T.: Semantic retrieval: multiple response model for context-aware learning services. Int. J. Inf. Tech. Commun. Convergence **42**(3), 253–267 (2012)
23. Tortorella, R.A., Graf, S.: Personalized mobile learning via an adaptive engine. In: 2012 IEEE 12th International Conference on Advanced Learning Technologies (ICALT), pp. 670–671. IEEE (2012)
24. Weiser, M.: The computer for the 21st century. SIGMOBILE Mob. Comput. Commun. Rev. **3** (3), 3–11 (1999)
25. Schilit, B., Adams, N., Want, R.: Context-aware computing applications. In: Workshop on Mobile Computing Systems and Applications, pp. 85–90 (1994)
26. Vandewaetere, M., Vandercruysse, S., Clarebout, G.: Learners' perceptions and illusions of adaptivity in computer-based learning environments. Edu. Tech. Res. Dev. **60**(2), 307–324 (2011)
27. Schill, A., Springer, T.: Verteilte Systeme. Springer, Berlin (2007)
28. Dey, A., Newberger, A.: The context toolkit (2000). http://contexttoolkit.sourceforge.net/
29. Yamabe, T., Takagi, A., Nakajima, T.: Citron: a context information acquisition framework for personal devices. In: IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (2005)
30. Bernados, A., Tarrío, P., Casar, J.: Casandra: a framework to provide context acquisition services and reasoning algorithms for ambient intelligence applications. In: International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 372–377 (2009)
31. Maia, M., Andre, F., Benedito, N., Romulo, G., Windson, V., Andrade, R.: Loccam-loosely coupled context acquisition middleware. In: 28th Annual ACM Symposium on Applied Computing, pp. 545–541 (2013)
32. Fahy, P., Siobhan, C.: Cass—middleware for mobile context-aware applications (2004). http://www.sigmobile.org/mobisys/2004/context_awareness/papers/cass12f.pdf
33. Hofer, T., Schwinger, W., Pichler, M., Leonhartsberger, G., Altmann, J., Retschitzegger, W.: Context-awareness on mobile devices—the hydrogen approach. In: 36th Annual Hawaii International Conference on System Sciences (2003)

34. Román, M., Hess, C., Cerqueira, R., Ranganathan, A., Campbell, R., Nahrstedt, K.: Gaia: A middleware infrastructure for active spaces. SIGMOBILE Mob. Comput. Commun. Rev. **6**(4), 65–67 (2002)
35. Hong, J.: The context fabric: an infrastructure for context-aware computing. In: CHI'02 Extended Abstracts on Human Factors in Computing Systems (2002)
36. Cheb, H., Finin, T., Joshi, A.: An intelligent broker for context aware systems. In: Adjunct Proceedings of Ubicomp, pp. 183–184 (2003)
37. Chen, G., Kotz, D.: Solar: an open platform for context-aware mobile applications. In: Proceedings of the First International Conference on Pervasive Computing, pp. 41–47 (2002)
38. Gu, T., Pung, H., Zhang, D.Q.: A middleware for building context-aware mobile services. In: IEEE 59th Vehicular Technology Conference, 2004. VTC 2004-Spring, vol. 5, pp. 2656–2660 (2004)
39. Bardram, J.: Tutorial for the java context awareness framework (jcaf), version 1.5 (2005). http://www.daimi.au.dk/~bardram/jcaf/jcaf.tutorial.v15.pdf
40. Leach, P., Mealling, M., Salz, R.: RFC 4122—A Universally Unique IDentifier (UUID) URN Namespace (2005). http://tools.ietf.org/
41. Silverman, W.L., Forum, L.: Learning and teaching styles in engineering education. Eng. Edu. **78**(June), 674–681 (2002)
42. Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0, Abschlussbericht des Ar-beitskreises Industrie 4.0, AGacatech – Deutsche Akademie der Technikwissenschaften e.V., April 2013, http://www.bmbf.de/pubRD/Umsetzungsempfehlungen_Industrie4_0.pdf (ARD Studie 2014) Online Medienutzung 2014; http://www.ard-zdf-onlinestudie.de/index.php?id=483)
43. Julian, D., Ulrike, L.: An infrastructure for cross-platform competence-based assessment. Accepted for: CHANGEE 2015. Facing the Challenges of Assessing 21st Century Skills in the Newly Emerging Educational Ecosystems. Toledo, 18 Sept 2015

http://www.springer.com/978-3-319-26516-2

Mobile, Ubiquitous, and Pervasive Learning
Fundaments, Applications, and Trends
Peña-Ayala, A. (Ed.)
2016, XII, 223 p. 56 illus., 44 illus. in color., Softcover
ISBN: 978-3-319-26516-2