"My Environment" – A Dashboard for Environmental Information on Mobile Devices

Thorsten Schlachter[1], Clemens Düpmeier[1], Rainer Weidemann[1], Wolfgang Schillinger[2], and Nina Bayer[2]

[1] Karlsruhe Institute of Technology (KIT)
Institute for Applied Computer Science
Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany
{thorsten.schlachter,clemens.duepmeier,rainer.weidemann}@kit.edu
[2] Baden-Wuerttemberg State Institute for Environment, Measurements and Nature
Griesbachstraße 1, 76185 Karlsruhe, Germany
{wolfgang.schillinger,nina.bayer}@lubw.bwl.de

Abstract. This paper describes a universal approach to the development of a cross-platform and multi-functional environmental mobile application. According to the Pareto principle [1] only common use cases are implemented and can be described in a lightweight description format instead of being explicitly programmed. These use cases include information about the environment ("my environment"), reporting of environmental data ("crowd sourcing"), and environmental experience ("electronic nature guide").

1 Introduction

Every month users of mobile devices are facing many new applications. Those can be used to access information of their interest in almost any place and situation using context information such as current position and time. People use apps for localized weather forecasts, travel information (including delays), news, stock quotes, etc., and maybe even provide information by themselves, implicitly by allowing apps to send location and speed (traffic flow) information back to service providers or explicitly by reporting their own observations via a crowdsourcing application.

At least in Europe authorities are committed to actively provide environmental information to the public [2]. For that reason many environmental information systems (EIS) do implement web frontends, web services, or even service oriented architectures and it is relatively easy to use these for the implementation of mobile frontends, e.g. mobile apps.

The range of environmental information provided varies from unstructured data, like text documents and media files, up to structured and semi-structured information, like tabular data, spatial data, metadata, time series, charts, and also includes specialized information for specific applications. Nevertheless, although data is very diverse some recurring usage patterns can be observed.

In [3] and [4] a descriptive approach of a universal mobile application accessing EIS based on extended OpenSearch descriptions, outlined in section 2, was presented.

In [5] and [6] we added our first lessons learned in the field of cross platform development for mobile devices, summarized in section 3.

Within the rest of this paper we would like to add our ideas of an integrated display of environmental information for our app. We additionally present an extension of our use cases, now including environmental reporting and environmental experience for both, the public and professionals.

2 No Programming Required!

In order to avoid programming for new fields of application the most important concept for this environmental app is a description format, called "target system description" based on the OpenSearch description format. The OpenSearch description format allows for the description of URL patterns for dynamic access to web systems. Originally the OpenSearch descriptions format was designed for access to web interfaces of search engines and mainly supplies a syntactic description for such queries. The semantics of its parameters is rigid. However, OpenSearch descriptions can be extended. For the purpose of use for a mobile environmental application, the following features have been added:

- Freely definable URL parameters.
- Semantic classification of these parameters, i.e. a rudimentary description of their semantics, for instance, the geographical position of a place in a certain format.
- A description of the sources of these parameters, e.g. automatically determined from a GPS component or by user selection from a given list of values.
- Structural presentation of the results, e.g. mapping of attributes.
- (HTML-)Templates for the display of certain result types.
- Content-dependent grouping information.

Such extended OpenSearch descriptions supply the necessary information for the registration of an information source in an application, for the generation of queries, and for the visualization of received data.

The template mechanism can also be applied to HTTP POST requests, which enables the app to write data to various destinations, e.g. REST services. With added HTTP basic authentication as well as OAuth (2.0) authorization flexible services for the storing of reported data can be described.

A "target system catalog" format, which extends the RSS 2.0 specification, is used for the description of multiple target systems, allowing the distinction of target systems, e.g. for different customers (Federal States) using the same app, and also the grouping of related target systems, e.g. different types of nature reserves or conservation areas.

3 HTML5-Based Cross-Platform Approach

Instead of developing independent native apps for each particular target system (possibly with the use or consideration of the appropriate specification), cross platform development aims to deploy one application without (or at least without major) changes to multiple target platforms.

This is often done by combining an unchanged common core of the application (top box in Fig. 1) with platform-dependent parts provided by cross development frameworks.

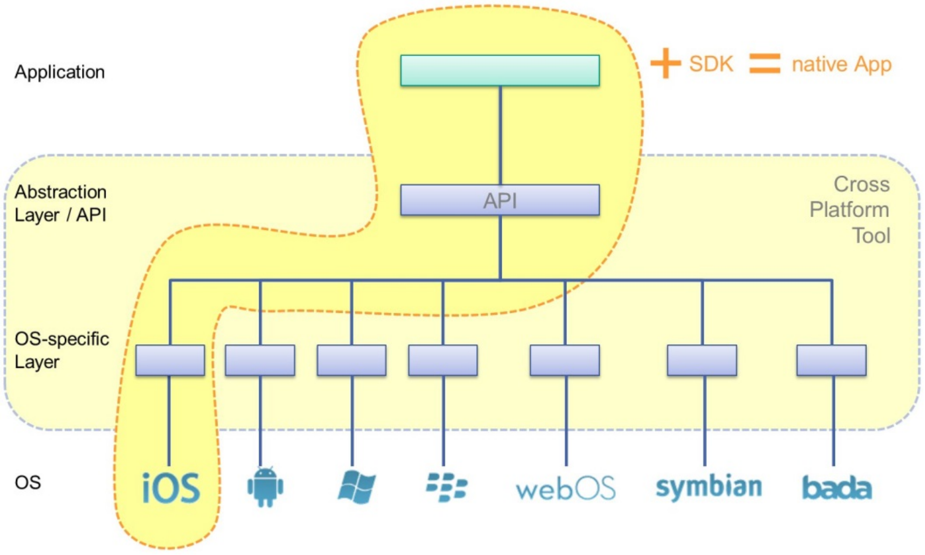

Fig. 1. Deploying an iOS-application using a cross-platform development tool

Such a combination then can be delivered for a particular platform. In Fig. 1 the OS specific "package" of an application, platform-independent and platform-dependent parts, is shown by the darker highlighted area.

The cross-platform development tool (brighter highlighted area) provides an application programming interface (API) against which the application can be programmed. On its backend it provides a specific implementation for each target platform (OS) that translates abstract commands into native functionalities of the system. According to the respective programming interface manufacturers of cross-platform development tools have to implement the OS-specific layer for each supported operating system once. This is especially easy if well-established features, commonly available for all or at least many target systems, have to be implemented. More specific features may be emulated by the interface, in the worst case, they are not supported.

Some manufacturers of cross platform development tools fall back on pre-existing technologies. One possible technology of note is Java, which is available for all popular mobile platforms.

A similar approach relying on widely available technologies is based on Web technology. The central idea is to run the actual application within a Web browser component which is already available on all platforms.

With the help of a cross platform Web development tool (and possibly using a platform-specific software development kit (SDK)) the generated native app contains all of the necessary resources for the application, in particular at least one HTML page, any associated graphics and CSS files as well as one or more JavaScript libraries that contain the abstract interface and the OS-specific implementations of the API. The display of the graphical user interface creates the browser in the form of HTML components, whose appearance can be controlled using Cascading Style Sheets (CSS).

The application logic is implemented with the help of the libraries in the form of JavaScript functions that may also provide functionality that is not standard in many Web browsers such as Multi-touch controls.

After a first prototype for the Android platform was fully implemented in Java, a second prototype based on Web technologies (HTML5) was developed to compare native and cross platform approaches. The second prototype was able to proof that a cross platform approach for the app is more feasible and so the first productive version of the app is based on Javascript-based Web technologies and a cross platform toolkit providing native functionalities through a common Javascript API.

The used PhoneGap/Cordova framework for cross development supports the particular use of components of different operating systems (including iOS, Android, Windows Mobile and BlackBerry) in a portable way. I.e. PhoneGap is able to access the camera, contacts, the file system, accelerometers, and GPS.

JQuery Mobile (JQM) provides a fully touch-optimized and asynchronous interaction with mobile browser-based applications. JQM runs on all commonly used browsers (like Firefox, Chrome, Opera and Safari), but in detail the display may show browser- and platform-dependent differences.

The Google Maps API is used for the display of maps and other data which include geospatial representations.

4 An Environmental Dashboard

The first prototypes of our mobile application behaved like a collection of single mini applications for querying EIS. Data provided by one data source has been displayed after a user request and has not been put in context of information provided by other data sources.

Our actual approach combines data and displays from several data sources within a kind of dashboard, dynamically changing when context changes or data sources provide new data (Fig. 2). The user may register for certain types of information and/or certain data sources. The app dynamically queries these using context information like sensor data (GPS, radio cell) or default settings provided by the user.

For example, spatial data from multiple data sources are merged into a single map view with different layers for each data source and the user has the possibility to individually switch single layers or groups of layers on or off, and thus display only the relevant data important to him.

Other areas of the dashboard can show other types of information, e.g. a media carousel can combine previews of all media objects collected from several data sources, and there still may be areas for other specialized mini apps, e.g. display of water levels or air pollution and the respective forecasts.

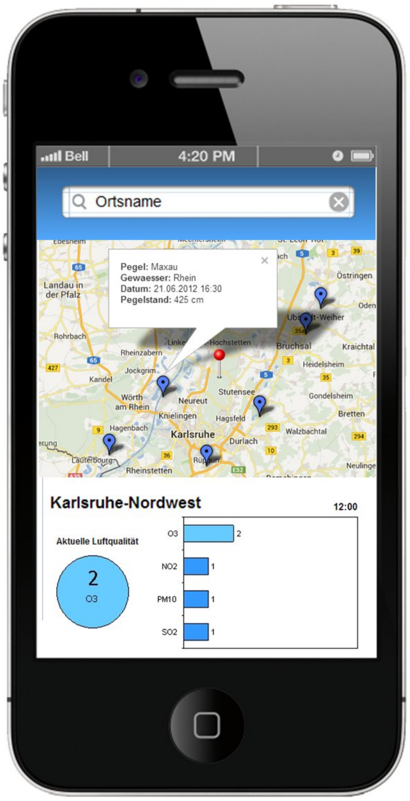

Fig. 2. Environmental dashboard with preview map and air quality data

Starting at the dashboard the user has additional possibilities to navigate to a more sophisticated query form, a more detailed display of a single item, or even to the original information system. We believe that in most cases a quick overview with a reduced complexity at the dashboard will meet the user's needs. I.e., a user frequently may be interested in the actual ozone levels before going jogging. With sports in mind he rarely is interested in querying a time series of ozone data at his location, even if he could do so.

Nevertheless the user is provided with functions to formulate complex queries or to filter larger hit lists. The navigation philosophy "dashboard/map – search/filtering – detail view – alarm/navigation/reporting" is applied within all use cases of the app. Key views and click paths are shown in Fig. 3.

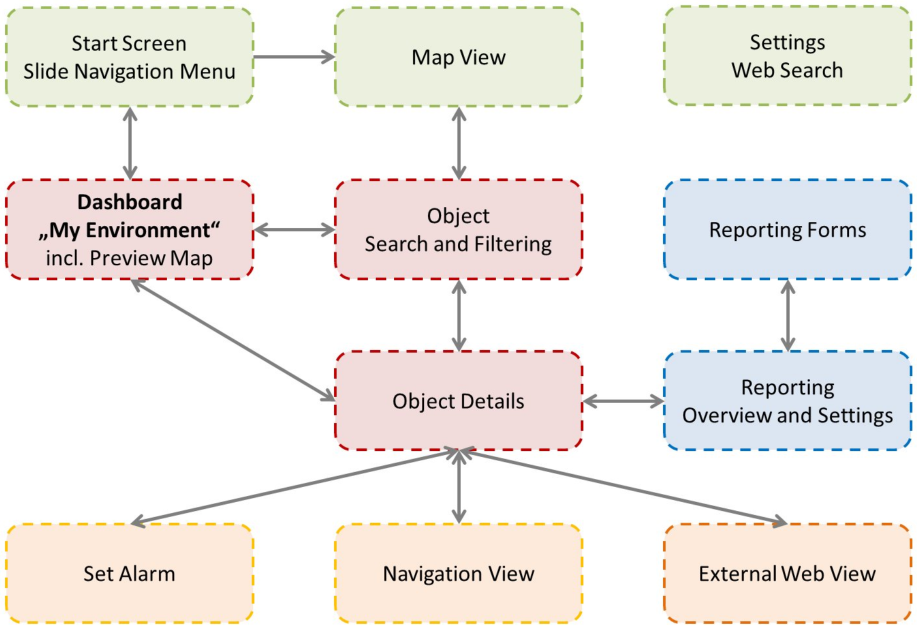

Fig. 3. Key views and click paths for the use cases "information" and "reporting". (Views for the use case "environmental experience" are not included.).

5 Beyond "Pure" Environmental Information

In addition to the environmental information dashboard, the app implements two other use cases, "reporting of environmental information" as well as "environmental experience" (electronic nature guide).

In the majority of cases reporting of environmental information follows a pattern of two. "Ad hoc reporting" aims to capture spontaneous observations, i.e. to take some photos or a video of a certain species, tag them with some additional information (e.g. locality, timestamp and some user input), and upload them to a server. In contrast, "systematic reporting" mostly does not start from scratch but aims to verify or complement existing information. This means a user has to be enabled to query (search and filter) existing objects, to view object details, and to supplement these details. For example, an employee moves to multiple measuring points and checks their condition. So he may be provided with additional routing information, and, of course, with the existing data for each station. In both cases there may be need for caching collected data offline on the device and for a delayed upload (to an EIS or to the cloud).

Many functions of the app are implemented using cloud services. For example, routing information is provided by the Google Directions API [7], and place names are determined using the Google Geocoding API [8].

From the perspective of EIS cloud services can help to close the gap between available and published data by providing infrastructure, storage, services, or even whole applications. For example spatial data easily can be (re-)published using tools like Google Maps for Business [9] when original systems do not provide standardized interfaces or are not powerful enough to cope with a large number of simultaneous requests. Structured or binary data may be stored in the cloud using services like Google Fusion Tables or Google Drive and the respective APIs [10, 11]. The discussion of the use of cloud services will be subject of another publication.

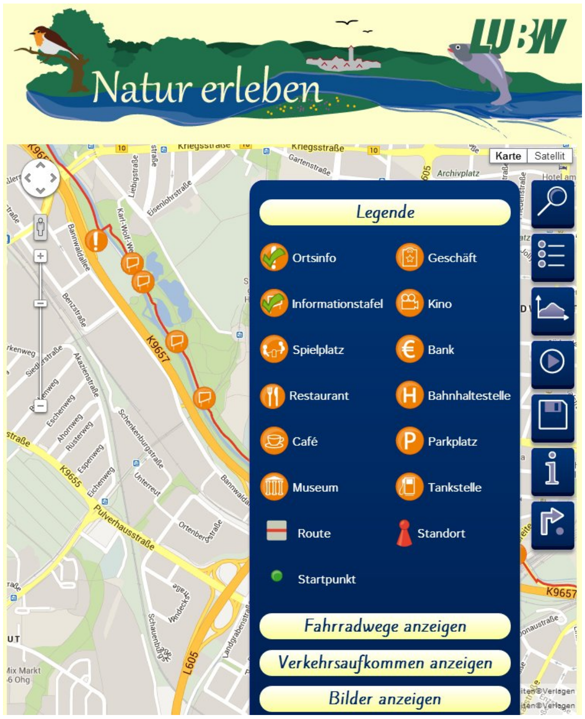

Fig. 4. Display of environmental and touristic information along a given track creating an environmental experience for the user. (Prototype by N. Bayer).

"Environmental experience" is our third use case, which has been prototypically implemented as a web application (Fig. 4) [12]. It combines aspects of environmental information and environmental reporting with the navigation along a given track such as a nature trail, as well as "learning by playing" elements in the manner of geocaching [13].

The application enables the user to navigate along the nature trails and provides him additional information (text, media, technical information). Besides fixed predetermined routes, there are also geocaching routes, where the user only knows the starting point of the route and is provided with basic additional information/criteria like the length of the course. The route itself is described by geocaching waypoints which arise from the repeated answering of questions.

6 Conclusion and Outlook

With three applications within a single app, description instead of programming, and cross platform capabilities our project “Meine Umwelt” (“My Environment”) has the capability to cover many standard requirements for environmental mobile applications.

However it has room for improvements in the future. At this time information is actively pulled from EIS. Push services should be enabled in order to avoid needless time and energy consuming requests.

The use case “environmental experience“ is still in an early stage and needs to be improved. And many more EIS have to be connected to the app so that the user can be informed comprehensively on all his relevant environmental issues.

References

1. `http://en.wikipedia.org/wiki/Pareto_principle`
2. Directive 2003/4/EC of the European Parliament and of the Council of 28, on public access to environmental information and repealing Council Directive 90/313/EEC (January 2003), `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32003L0004:EN:NOT`
3. Schlachter, T., Düpmeier, C., Geiger, W., Weidemann, R., Ebel, R., Tauber, M., Zetzmann, K.: Concept of a universal mobile application accessing environmental information systems. In: EnviroInfo 2011: 25th Internat. Conf. on Environmental Informatics, Ispra, IT, October 5-7 (2011)
4. Schlachter, T., Düpmeier, C., Geiger, W., Weidemann, R., Ebel, R., Tauber, M., Zetzmann, K.: LUPO mobil. Ein Schichtenmodell zur Auswahl und Nutzung von Umweltdiensten auf mobilen Endgeräten.“; In: Mayer-Föll, R. [Hrsg.] UIS Baden-Württemberg. F+E Vorhaben KEWA Phase VI 2010/11; KIT Scientific Reports KIT-SR 7586; pp. 33-42 (2011)
5. Schlachter, T., Weidemann, R., Ebel, R., Schillinger, W., Zetzmann, K.: Building Mobile Environmental Apps Using Web and Cloud Technologies. In: EnviroInfo 2012: 26th Internat. Conf. on Environmental Informatics, Dessau, DE, August 29-31 (2012)
6. Schlachter, T., Düpmeier, C., Weidemann, R., et al.: LUPO mobil - Nutzung von Webtechnologie zur Entwicklung plattformübergreifend einsetzbarer, mobiler Umwelt-Anwendungen“; In: Weissenbach, K. [Hrsg.] UIS Baden-Württemberg. F+E Vorhaben MAF-UIS Phase I 2011/12; KIT Scientific Reports KIT-SR 7616; pp. 59-70 (2012)
7. `https://developers.google.com/maps/documentation/directions/`
8. `https://developers.google.com/maps/documentation/geocoding/`
9. `http://www.google.com/enterprise/mapsearth/products/mapsengine.html`
10. `https://developers.google.com/fusiontables/`
11. `https://developers.google.com/drive/`
12. Bayer, N.: Konzept und prototypische Realisierung eines mobilen Umweltführers Bachelor thesis, Hochschule Karlsruhe – Technik und Wirtschaft (2013)
13. `http://www.geocaching.com`