**A mobile web ohmage application for data collection**  
Zorayr Khalapyan, Edward Sakabu, Joshua Selsky, Hongsuda Tangmunarunkit, Rose Rocchio, Deborah Estrin  
  
**Abstract**  
Advances in mobile devices and web technologies have enabled users to pull and display online data from the cloud almost anywhere and anytime. At the same time, the same technologies can enable phone users to observe, collect and publish in-situ personal or community-related data to the cloud. In this paper, we describe our web-based mobile application, Mobile Web ohmage Client (MWoC), that can be used to record in-situ data entered by the user anywhere and anytime. The data will be uploaded and published to ohmage---an open-source, mobile to web platform that records, analyzes, and visualize both human-in-the-loop in-situ data, and continuous data streams passively collected from mobile devices. MWoC is a lightweight single-source and platform-independent mobile application and therefore can be run on a broader platforms compared to the more comprehensive ohmage android application. MWoC utilizes the Mobile Web Framework (MWF) for a robust cross-platform presentation and PhoneGap for accessing native device features. We will present our requirements, design, lessons learned and future direction in this paper.  
  
**Introduction**  
Advances in mobile phone technologies, the ubiquity of mobile phones and expansive cellular service areas enable mobile phones to be increasingly used as tools to observe and understand the patterns, habits and environments of individuals and communities. Ohmage is an open-source, mobile to web platform that records, analyze, and visualize data from distributed mobile devices. Ohmage supports two data stream types: 1) prompted experience samples which are small scriptable survey forms for users to enter data in the field, i.e., in-the-moment experience; and 2) continuous streams of data passively collected from sensors or applications on-board the mobile device (e.g., location traces).   
  
Ohmage users can quickly create a data campaign which is a collection of surveys to be answered by participants. [prompt types]. Researchers can also specify conditional branching of questions based on users' answers. Newly defined surveys are uploaded via a web browser and automatically become available for participation. To participate, a user login to the application and downloads the particular campaigns(s) in which they will participate. Surveys can be prompted based on time/location reminders or initiated by the user. All submitted responses, along with continuous data streams are stamped automatically with time and geographic location and automatically uploaded to the ohmage server for further processing, analysis and visualization. Via web browsers, researchers and users are able to access and visualize collected data and their analyses in real-time.   
  
  
**MWoC requirements**  
The first version of MWoC is designed to be light-weighted and focuses around the data capture of prompted experience samples with time-based reminders. These features are:    
\* Authentication. Once a user login, MWoC will authenticate the user with the ohmage server. Upon successful authentication, MWoC will store a long-term token locally for subsequent communication to the ohmage server. All user data including the long-term token will be deleted after the user logout.  
\* Data capture of prompted experience samples. Users interacts with MWoC to browse through a set of available campaigns and download ones that they want to participate. The downloaded campaigns will be stored locally on the phone for better responsiveness and offline support. To complete a survey, MWoC goes through the campaign definition (written in XML) and renders appropriate sequence of prompts (i.e. question) based on their types (e.g. menus, number, time, free text, image, etc.) and condition. Upon survey completion, all survey responses will be timestamped, geocoded and uploaded to the ohmage server. In the case of network outage, it will store survey responses locally on the phone and will try to automatically upload the data upon future launching of MWoC. Users can also perform manual upload of the data. To support in-situ data capture, MWoC needs to function while the phone is offline.    
\* Time-based reminder. MWoC allows users to configure time-based reminders to be triggered during specific time and condition (e.g. send a reminder to complete a survey unless the survey has been completed within the last 2 hours). Upon reaching triggered criteria a notification in the form of ring tone or vibration will be generated to remind users to complete their survey(s).  
  
**Design**   
[insert a diagram here]  
  
To support single-source and platform-independent mobile application, MWoC is developed based on web-based technologies (such as HTML5, CSS, JavaScript) and PhoneGap[]. MWoC consists of three different components:  
  
1.  Presentation. MWoC utilizes the Mobile Web Framework (MWF)[] for mobile-based presentation. MWF is a cross-platform web framework that provides a robust presentation layer that allows web applications to define a single set of markup optimized for HTML5 capable mobile devices. MWF focuses on mobile web standards, semantic markup, device agnosticism and graceful degradation which enables developers to create mobile-styled web applications with relative ease of use. MWF is a client-server architecture. MWF client heavily relies on MWF server for the most compatible CSS and HTML content. To provide offline support, MWF server is wrapped in a container and stored locally on the phone using PhoneGap (described below). MWoC currently utilizes MWF for menu, title, content and buttons.   
  
2. Business Logic. MWoC business logic is implemented purely in JavaScript. It relies on HTML5 local storage to store dynamic content such as campaign lists, campaign  configuration XMLs files ,survey responses and as well as user-captured images. It also relies on the Native Device Adapter to access native resources such as files and camera (described below). MWoC dynamically generates individual pages to render different prompts based on the XML description. The dynamic pages are implemented as HTML5 forms [Zorayr checks this] and are stored in the local storage for users to go back and forth and change their answers before submission. Once the user clicks submit, MWoC will create a survey response and associate it with a timestamp and GPS location, then submit the survey response to the server. If the phone is offline, the survey response will be added to the upload queue stored in the local storage for future upload attempt. To make the upload seamless to the users, MWoC implements an automatic upload of survey responses in the upload queue upon the on-click event of the MWoC application.  
  
3. Native Device/Application Adapter. PhoneGap provides a set of APIs for web applications to access a set of phone devices such as camera, file, geolocation, notification, etc. It also allows   
  
PhoneGap is a HTML5 application platform that provides supports for native applications developed with web technologies (i.e. HTML and JavaScript) to get access to the native APIs and application stores. PhoneGap provides APIs to a set of phone devices such as camera, file, geolocation, notification, etc.  
  
PhoneGap will use standards-based web technologies to bridge web applications and mobile devices. It is used to contain static web pages; JavaScript, CSS and HTML files, as well as images locally on the phones. [Zorary: provide examples of these web pages].   
  
**Lessons Learned**  
\* asynchronous file upload to the server (with images): problems with sth to do with iforms?  
\* Cross-domain access restriction  
\* responsiveness issue: on-click vs on-touch (click handling and touch handling)  
\* Time-based reminders: Explore localNotificationPlugin on PhoneGap.  
\* Notification system:  
\* Orientation problem  
\* Stateless page loads. There is no mechanism to allow passing an already created JavaScript object (e.g. campaign configuration) to a new loading page (e.g. survey page). Hence, to display a specific survey, the entire campaign XML file has to be loaded and parsed again.To address this issue, we adopted an Event-Driven Object Construction pattern which is to have the child object wait to instatiate itself only after the parent object s constructed.    
  
**Future Work**   
MWoC and ohmage will be deployed in a behavioral research study of health and HIV-transmission risk behaviors of gay men. MWoC will also be used as an option in to teach high schoolers in LAUSD on data collection and analysis.  
  
\* Feature improvement  
\*\* Visualization and feedback  
\*\* Context-based notification  
\*\* Automatic upload: Ideally, all data should get uploaded automatically without users's invovelement. To do this, we need the capability to run a background process. However, IOS doesn't explicitly support background process.  
  
**References**  
[] ohmage  
[] MWF  
[] PhoneGap