Firepoint: Porting Application to Mobile Platforms

Artem Timonin, Artem Kalinin, Alexander Troshkov, Kirill Kulakov
Petrozavodsk State University (PetrSU)
Petrozavodsk, Russia
(timonin, kalinin, atroshko, kulakov)@cs.karelia.ru

Abstract

Forest saving is one of important human problems. Forest fires are on the list of global natural disasters every year causing enormous damage to the environment. The Firepoint application provides to the user information on fire detection, obtained from the FIRMS service. First version of application was implemented for Harmattan platform.

This article is about porting Firepoint to three different platforms: Android, Symbian (Anna and Belle) and Nokia S40 Touch and Type. Source code for Android and S40 had been started from scratch. Nevertheless some modules had been saved and source code had been changed from C++ to Java.

Index Terms: Notification, Fire, Qt, Firepoint.

I. Introduction

Every day mobile platforms increases popularity. People uses mobile devices to play the games, communicate between themselves, make some media objects and so on. Many companies like Apple, Google, Microsoft and Nokia provides mobile platforms for devices. Most platforms based on Linux kernel, but has own environment, programming tools and kits. To distribute an application to many users we need to provide the version of application which is support selected platform.

The one approach is to use cross-platform frameworks, for example, Qt or HTML5. This way has several restrictions. The cross-platform framework doesn't have ability to use native calls and therefore application has slow speed [1]. Also some platforms, like Nokia S40 "Touch and Type" didn't support popular cross-platform frameworks.

The another approach is to rewrite the core keeping architecture solutions. This way is more expensive, because we need to rewrite source code from one platform language to another. But in this case we can access to native programming interface and can make application more conventional. This paper describes last approach for porting the Firepoint application [2].

Firepoint provides to the user information on fire detection, obtained from the FIRMS service [3]. Nearby fires are displayed on the map. The location of the nearest forest areas is displayed to the user. Also application has the possibility of connection with the emergency services for fire detection and fire warning.

The project was initiated in 2011 at Petrozavodsk State University (PetrSU), FRUCT laboratory of wireless and mobile technologies. Firepoint belongs to the family of FRUCT research projects [4]. The project is co-funded by grant KA179 of Karelia ENPI programme, which is co-funded by the European Union, the Russian Federation and the Republic of Finland.

The first version of application was implemented to Harmattan platform on Qt/QtQuick platform using QtMobility to positioning and showing maps [5]. Then the application ports to three different platforms: Android, Symbian (Anna and Belle) and Nokia S40 "Touch and Type". Source code for Android has been started from scratch because we are found some problems of porting Qt applications to Android platform [1]. Nokia S40 doesn't have any Qt ports. Nevertheless some modules have been saved and source code has been changed from C++ to Java. This is due to different paradigms, patterns of user interface and many other things.

The rest of the paper has the following structure. Section II describes common architecture and models, which are remained the same. Also this section contains some notes about porting application to Symbian platform. Section III describes porting Firepoint to Android platform. Section IV presents porting Firepoint to Nokia S40 platform. Section IV describes changes between Harmattan and Symbian versions of Firepoint.

II. Application funcions and architecture

Application provides the following functions.

- Show fires on the map. Application loads points of fires from FIRMS service for all world or selected region and shows it on map. User can browse points. If some place has many points of fire then points are group by cluster. The main feature is that all data can be load before trip to forest when we have stable Internet connection. During the trip application uses cached data.
- Alerts user about nearest fire. If application detected than user is near fire point then alert message will be shown.
- Emergency call. If user found new fire in the forest, then application shown current coordinates of user and provides ability to emergency call.
- Preservation forests from fires. Application contains two comics which are described how to behave in a forest: “What to do in case of fire” and “How not to cause a fire”.

High-level architecture of Firepoint application is presented in Fig.1. Main module starts application and initiates communications between other modules. User interface shows nearest fires and forestry corresponds user position as a layers in Map pattern. Also User interface contains settings form, comics and interface for emergency calls. Fire module obtain data from FIRMS service, clustering and produce “fire layer” corresponding to the user position. Forestry module loads forestry points from local file (we assume that the forestry vary rare) and produce “forestry layer” corresponding to the user position. Positioning module detects user position throw GPS or GLONASS system or Yandex.Locator system if Internet connection is available.

In ported versions we have the following differences.

Each platform has guidelines for developing user interface and Firepoint has been created in compliance with them. Main difference between Harmattan and Symbian versions was creating flexible UI because there are many display sizes on Symbian devices. Symbian version has more dialogs and some changes in preferences for Location detection which has been added during publication in Nokia Store. So, for user these versions are very similar.

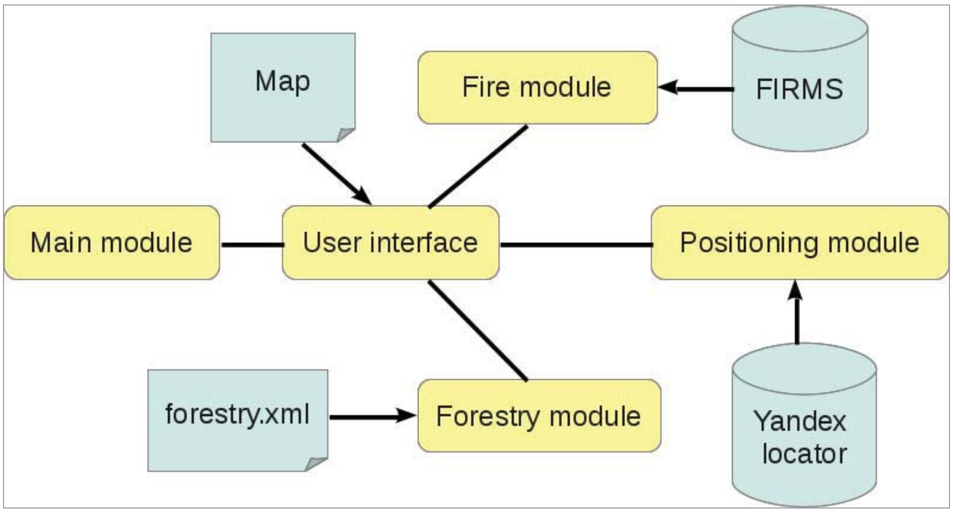

Fig. 1. Firepoint architecture

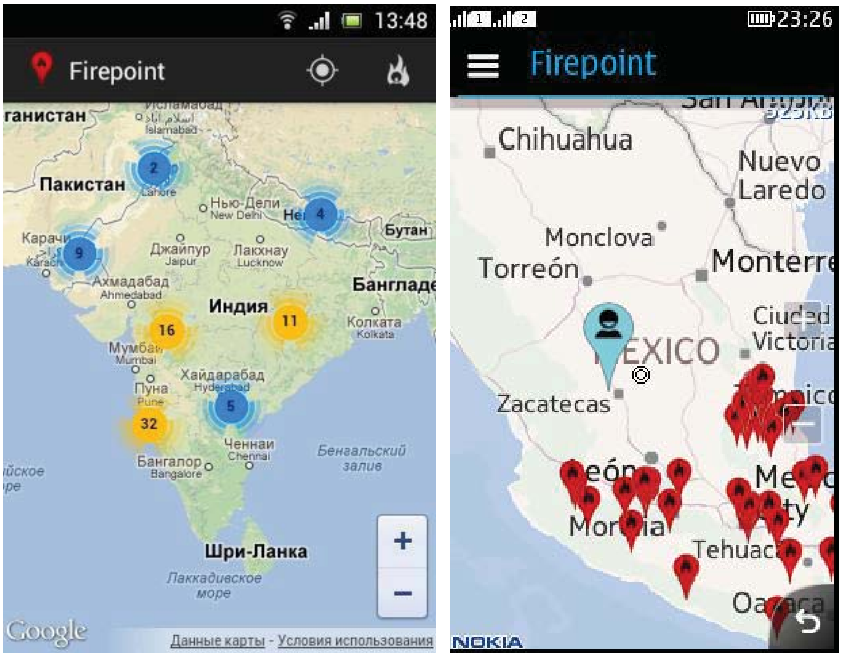

Fig. 2. Screens of Firepoint in Android and Nokia S40

User interface versions for Android and Nokia S40 have a lot of changes in comparison with Harmattan and Symbian versions (see fig. 2). For example there is no menu page in these versions because application has a tool bar at the top of screen which provides all functions. Also devices on Android have a hardware settings button. Of course Firepoint for Android uses native graphic elements. That’s why style is not similar to Harmattan version. Version for Nokia S40 differs from other versions by “back” button in the bottom right corner. It is system feature because there is no hardware button in full touch devices.

And there are some useful icons on tool bar in Android versions but tool bar in version for Nokia S40 hasn’t such function.

The next difference is a implementation of points clustering function. For Symbian platform we uses original source code without any modifications. In Android platform points clustering has several implementations in libraries built on official Maps API. Most of these libraries was created before API version 2 has been released, therefore it should be rewritten to use with a newest version of Maps API.

Polaris [6] is one of the libraries that have a clustering possibility (only in experimental branch). The clustering is not the main mission of this tool. Polaris is not suitable for a big amount of points. For example, building clusters from 2000 map points will take about 20 seconds. It is absolutely improper from the end user point of view. Android Maps Extensions [7] is ready to use in API v.2 library. It implements dynamical grid-based clustering that works absolutely without stumbling the map object and the UI.

Unfortunately, we don't found any implementation of clustering functionality for Nokia S40 platform. We decide to disable loading points for all world.

III. PORTING TO ANDROID PLATFORM

Firepoint for Android has been written natively, using Android SDK [8] and Google Play Services [9]. The most problematic part was Map subsystem. It has been written two times: using WebView (as in original Firepoint for Harmattan) and using official Google Maps Android API.

The easiest way to implement the Map was to copy it from original Firepoint for Harmattan. It implemented using HTML and Google Maps Javascript API. Unfortunately, WebView on Android works very slow with a lot of complicated objects: map, points and clusters of points. For comparison, N9 with Harmattan (1 GHz CPU and 1 Gb RAM) have no lags rendering map with a lot of points. And the two Android devices (1GHz CPU and 512Mb RAM; 1.5GHz CPU and 1Gb RAM) haven't provide any lagless user experience. Even just a map without points awfully lags on Android. For writing something with WebView, you should implement special interfaces from Javascript to Java. To call Javascript function from Java, you should call loadURL() method for your WebView object (e.g., webView.loadURL(“javascript:centerAtCurrentPosition()”);). Sending data from Java to Javascript takes a lot of time and it is the only one way to recieve points in HTML. Also, calling WebView methods from non-UI thread is deprecated. In the future it won't be not available to transfer a lot of data without sticking the UI.

In version of application with using official Google Maps Android API (part of Google Play Services) map works faster, but have one big issue: there are no any clustering capability. No matter, map works fast (except some lags on zoom level 0 and a big amount of points).

The source code was full writing again except one module which calculates distance between two points on a sphere (approximately, Earth is a sphere). If calculated distance is small enough, Firepoint notificates user about nearby fire.

There is a new way to check for points update: check 'Last-Modified' field in HTTP-header of remote file with fire points. This way is more intelligence, than in Firepoint for Harmattan (just ask user twice a day). FIRMS (Fire Information for Resource Management System, organization that provides fire data) updates file two-three times a day.

Initially Firepoint have been developing only for Android v.3.0 and later. But the market share of Android under v.2.3 is more than 50% (45.4% for 2.3.x and 8.1% for 2.2; Android Dashboards for 20.02.2013). This is due to acceleration of development: it is faster to implement an application for highest version of OS and then backport it than to track older versions compatibility. When you're writing for old versions, you are not only to write a code, but need to perform every-time checks for compatibility. Moreover, Android had radically changed the user interface (since 3.0) and Firepoint provides (by using libraries) identically one user experience on different versions.

The porting Android to older versions can be done in two ways: using Google compatibility library or using community libraries (like ActionBarSherlock, Holo Everywhere etc). First public version of Android has been ported with ActionBarSherlock [10] and the current version is ported using official compatibility library. There is a big difference between these two ways: official library doesn't save the UI style of Android 4.0 on older versions and ActionBarSherlock does (properly, it is the main idea of this library) and adds an extra weight to installation package).

IV. Porting to Nokia S40 platform

Beginning of development for Nokia Asha can be preceded in two different ways. First is native application traditional for mobile development. Second is web application relatively new for mobiles. They use different tasks: java apps — native (games, communication, productivity tools, multimedia), web-app (news, info, guides, social networks, brands). At first sight faster and convenient develop our application with web. Some big disadvantages of this way have been detected after little research. It isn’t possible to test application on device. It can be installed only after publishing in Nokia Store.

The main part of functionality of application is associated with the map. It should work quickly and correctly. It is possible to group points on the map in the most implementations of popular maps (Google, Javascript frameworks). There is no such possibility in Nokia for JavaME. It requires to group them by yourself. There are a lot of similar functionality to do. It increases development time.

V. Conclusion

Implementation takes 6 mounts and proceeded by three developers. The average time of developing for each platform (from scratch to end-user version) is about 3 months. S40 is the most code-comprising platform and Symbian (as an heir of Harmattan) is the most UI code-comprising. Source code metrics are shown in Table I. The size of source code for Symbian is similar of size of code for Harmattan.

All applications are tested in platform emulators and available devices. For Symbian we use Nokia Remote Device Access [11].

The version for Android is the most ease to distribute platform because of absence of any premoderation or strict rules, but for Symbian and Nokia S40 we have additional testing by Nokia Store team.

References

[1] Kirill Kulakov. Practice of Porting Qt applications to Android platform // Proceedings of 12th Conference of Open Innovations Association FRUCT (FRUCT12) — Saint-Petersburg: SUAI, 2012. — Pp. 268-271.

[2] “Firepoint in Launchpad”, Feb. 2012. [Online]. Available: https://launchpad.net/firepoint

[3] “Fire Information for Resource Management System”, Feb. 2012. [Online]. Available: http://firefly.geog.umd.edu/firms/

TABLE I
SOURCE CODE METRICS

Platform	Language	LoC
Android	Java XML	1076 813
Nokia Asha	Java ME	2108
Symbian	Qt QML	1004 3089

[4] “Open innovations framework program fruct,” Feb. 2012. [Online]. Available: http://fruct.org/
[5] Sofronova I., Timonin A., Kalinin A., Troshkov A., Borodin A., Kulakov K. Firepoint: Application for Identified Fires Notification // Proceedings of 11th Conference of Open Innovations Association FRUCT (FRUCT11) — Saint-Petersburg: SUAI, 2012. — Pp. 164-168.
[6] "Polaris", Feb. 2013. [Online]. Available: https://github.com/cyrilmottier/Polaris
[7] "Android Maps Extensions", Feb. 2013. [Online]. Available: https://code.google.com/p/android-maps-extensions/
[8] “Android SDK”, developer.android.com, 2013 [Online]. Available: http://developer.android.com/sdk/index.html
[9] Google Play Services, developer.android.com, 2013 [Online]. Available: http://developer.android.com/google/play-services/index.html
[10] ActionBarSherlock,actionbarsherlock.com, 2013 [Online]. Available: http://actionbarsherlock.com/
[11] Nokia Remote Device Access: http://www.developer.nokia.com/Devices/Remote_device_access/