The GPII on Smart Phones: Android

Javier Hernández Antúnez

Emergya, Sevilla, Spain
jhernandez@emergya.com

Abstract. The focus of this presentation is to go through all the aspects that are being covered during the works on the implementation of the Global Public Inclusive Infrastructure (GPII) [1] on Smart Phones, the scope, the status of the current implementation and upcoming developments where the Cloud4all [2] project is working on.

Since The Global Public Inclusive Infrastructure aims to become an international standard, one of the biggest challenges of the GPII project is to support all those devices that are using, and will use in the future, the technologies around the Smart Phones. This initial implementation is coming from the Cloud4all project, which has bet on the Android platform to demonstrate the features that the GPII will offer to us on every device that could run Android on it, either a Smart Phone, or Tablet, or DigitialTV, etc, and will serve as inspiration for future implementations on other Smart Phone platforms such as the popular iOS and Windows Phone, or the emerging Firefox OS, Tizen or Ubuntu Touch.

Keywords: Accessibility, Internet Access, Health, Social inclusion, Cloud, Mobility.

1 Background

The concept of Smart Phone was first used when Ericsson described its GS 88 "Penelope" concept as a Smart Phone, but nowadays, by Smart Phone we usually understand the technology who has brought the digital era of the Internet to our hands, as a tiny and lightweight, but a powerful device which can fit into our pockets.

For a long time now, several platforms that conceptually matched the definition of Smart Phone has existed such as Symbian, PalmOS, Windows CE, etc, and these triggered the competition between many companies for building the most successful one. As a result of this, today, around 1.5 billion of Smart Phones are being used around the world, where Android from Google, IOS from Apple, Windows Phone from Microsoft and BlackBerry, are the most relevant mobile platforms in the world.

2 Motivations

2.1 About Android

Although Android was developed early in 2005, it wasn't unveiled as a product until November 5, 2007, when a consortium of technology companies including Google,

C. Stephanidis and M. Antona (Eds.): UAHCI/HCII 2014, Part II, LNCS 8514, pp. 58–67, 2014.

device manufacturers such as HTC, Sony and Samsung, announced the creation of the Open Handset Alliance [3], with a goal to develop open standards for mobile devices. The first device running Android was announced the next year. Right now, Android represents more than the eighty percent of Smart Phones platforms on the market around the world.

As of the end of 2013, Android was the most popular operating system, with a 81.9% market share, followed by iOS with 12.1%, Windows Phone with 3.6% and BlackBerry with 1.8%. These numbers are reflected in the following table extracted from *Worldwide Smartphone Sales to End Users by Operating System* [4]

Table 1. Comparison table of Smartphone platforms and its Market Share

Operating System	2013 Market Share (%)
Android	81.9
iOS	12.1
Microsoft	3.6
BlackBerry	1.8
Bada	0.3
Symbian	0.2
Others	0.2

2.2 Online Software

Another concept which became popular at the same time we started using Smart Phones massively, was the ability to install software into a Smart Phone immediately, and this concept has impacted on the software business dramatically. And nowadays, we could call them:

- Google Play
- App Store
- Windows Phone Marketplace

By using these application markets, the people now have a huge amount of software available to them, and ready to be immediately installed on their Smart Phone devices.

Nowadays, the companies around the most successful application markets are making a profit of them, and talking in numbers, around 25$ billion in sales as we can read in this article [5]:

App stores run by Apple and Google Inc. now offer more than 700,000 apps each. With so many apps to choose from, consumers are estimated to spend on average about two hours a day with apps. Global revenue from app stores is expected to rise 62% this year to $25 billion, according to Gartner Inc,

2.3 Accessibility on Android

The Android community, and by having Google included into it, are working together to address the most common problems that people with disabilities find out when trying to use the system.

To illustrate the state of the art of the accessibility on Android, there is a detailed article about it from Darren Burton and Matthew Enigk called *Android Ice Cream Sandwich: Evaluating the Accessibility of Android 4.0* [6]. In this article, Darren Burton and Matthew Enigk go through the built-in features that Android 4 offers to address the problems of people with disabilities, taking special attention to low-vision and blind users. And as a resume of the most relevant features:

- A user who is blind or visually impaired does not need any sighted assistance to turn on the screen reader, so they can start using the device by their own
- Android comes with *Explore by Touch*, which allows to the user the ability to explore the content of the screen just by moving their fingers around the screen
- An accessible tutorial comes on screen when the screen reader starts, and the *Talk Back* speech synthesizer talks you through practicing how to use *Explore by Touch*
- The *Haptic Feedback* is a must have feature for the people who are blind or visually impaired and Android comes with it
- *Touch gestures* are well supported so users can use built-in, or define their own, gestures to handle the environment
- The *On Screen Keyboard* is accessible and usable within the screen reader
- Most of the basic but required tasks that we make on our Android device is accessible for people who are blind or visually impaired
  - Making a call
  - Messaging
  - Web browsing
  - E-mailing
  - Unlocking and answering an incoming call are easy

3 Goals

One relevant goal of the Cloud4all project is to create a real demonstration about how to cover the personal needs and preferences of a user who is using an Android device, either if it's a phone, a tablet, a DigitalTV, or a Kiosk running Android on it, and by using the benefits from the GPII personalization framework. But to specifically, cover all those needs and preferences that people with disabilities have when using an Android device, and solve their problems by automatically running ATs or any built-in feature.

From the Description of Work of the Cloud4all project we can extract the objectives to be addressed during the project, they are:

- To identify the built-in accessibility features of smartphones that can be auto configured based on the user profile. These accessibility features may include screen/graphical characteristics, user input and/or sound speech capabilities
- To design the accessibility solutions for mobile devices that will enable the auto-personalization from profile" (APfP) capability
- To build "auto-personalization from profile" (APfP) capability into prototype mobile devices. This solution provides an adaptative accessibility solution for smartphones that is based on the user profile and needs
- To test whether these disparate mobile environments can provide the same user experience for a common user profile. This will ensure the interoperability of the solutions and will allow users to change from one mobile device to another with minor effort

4 Implementation

The implementation of the GPII on Android is taking place on the Cloud4all european project, which is putting a lot of efforts on creating the mechanisms to take advantage of the GPII's "Auto-personalization from users' needs and preferences" features and capabilities.

4.1 Technical Solution

The implementation of the GPII on Android is being built on top of the GPII core architecture, and since the GPII core architecture is based on Node.js [7], this implementation makes use of a port of Node.js for Android, called Anode [8]. By using Anode, all the components from the GPII architecture can run locally into an Android device.

And like in the rest of the local implementations of the GPII (Windows and GNU/Linux), only a few platform specific parts were required to be implemented. All these platform specific developments are living together into GPII's Android repository on Github [9], and there are a lot of literature about the works on Android:

- An Overview of the details and information about the implementation in the wiki of the GPII project [10]
- A useful FAQ about this implementation [11]

To mention about the platform specific developments, the source code includes platform-specific modules that are required to communicate with Android's built-in features such as:

- Activity Manager, to deal with Android's activity manager and allow us to start or stop applications.

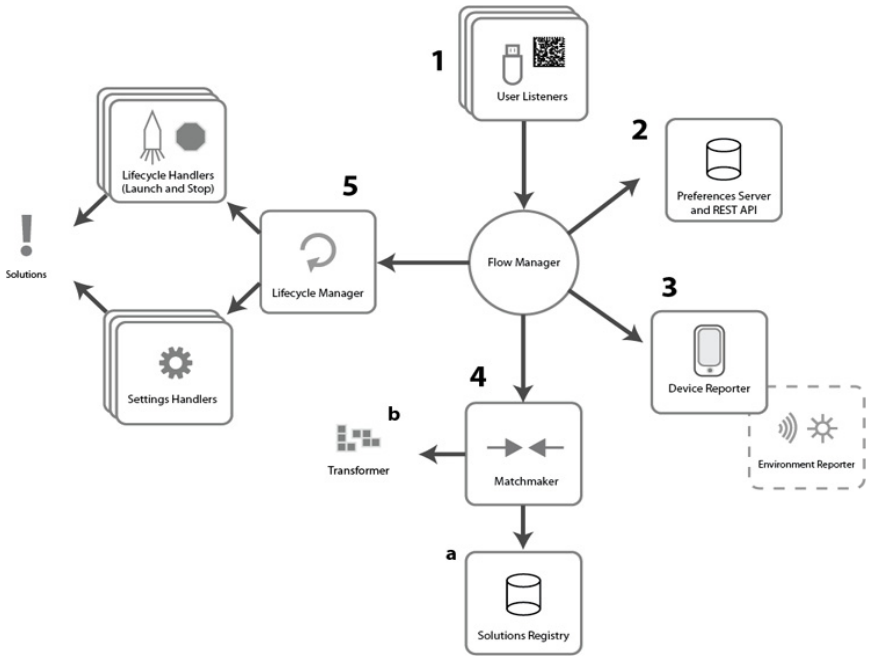

Fig. 1. GPII Architecture overview

- Android Settings, to deal with some settings that are accessed through a concrete API
- Audio Manager, to deal with system's volume levels
- Persistent Configuration, to deal with some settings that are accessed through a concrete API

All these components are loaded by the start script of the GPII on Android, and the work on this implementation ends up by registering the well-known Android's platform specific solutions into the GPII's universal source code repository. These well known solutions include:

- A description of a solution
- The settings handlers that the GPII uses to automatically configure this solution
- The available mechanisms to move specific settings from a platform or application into common terms [12]
- The lifecycle managers of each solution, describing how a solution has to be configured, started and stopped after logging out from the GPII.

Another platform specific work was in relation to the User Listeners. The User Listeners are the user's entry point to the GPII, and nowadays we support many ways to use the GPII on Android:

- By using an NFC tag
- By using a QR code

There are two different and working User Listeners, and they're available to download.

The first one [13] was developed by Tony Atkins and it includes support for NFC. The second one [14] was developed from some of the Cloud4all project partners, and includes the following features:

- Support for reading NDEF and GPII-specific mime-type NFC tags to log into the GPII
- Support for writing GPII-specific mime-type NFC tags
- Support for reading QR codes to log into the GPII
- Support for creating QR codes with a given user token

4.2 Developmental Issues

Since the beginning of the Cloud4all project, the Android team has been putting its effort on giving to Android (as a platform) the possibility to run the GPII core framework by itself, and this goal was successfully achieved by using Anode.

Despite all the restrictions regarding the access to the system's internals on Android, the implementation of the GPII on Android has the ability to take advantage of the auto-personalization from users' needs and preferences feature of the GPII, which has been addressed in the most recent versions of Android.

These problems and restrictions can be summarized in the following list:

- Root access are required to deal with some Android's internal components and APIs.
- Different versions of Android can provide a different set of settings that can be accessed through its APIs
- Debugging Node.js applications running on Android is hard
- Despite of the fact that Anode works, it still needs some improvements to make this implementation more rock-solid

Although these problems and restrictions are more or less under control, the Android team is continuously improving the system to address or at least, minimize the impact of having them.

4.3 Results

At this moment, the current implementation of the GPII on Android is a full-featured one, with the abilities to:

- Run their own instances of the GPII Architecture Framework by itself
- Intercommunicate with the system, configure its settings, and take advantage of the built-in features that Android has on it
- Intercommunicate with any Android's accessibility native service such as Talkback, the Android's built-in screen reader for visually impaired people

- Providing to third-party applications installed on the system the ability to take advantage of the GPII's "Auto-personalization from users' needs and preferences" features and capabilities, and being launched and automatically configured by the GPII
- Automatically translate settings from any other platform and or any application, into Android, and vice versa

5 Delivering the GPII to the End-Users

As part of Cloud4all, the implementation of the GPII on Android will be tested during the second pilots iteration of the project, where a lot of users with many different disabilities will have the opportunity to test this implementation, and to enrich the future developments on Android.

This second pilots iteration will take place this year again in three different pilot sites, in Germany, Greece and Spain.

As we can extract from the Cloud4all's deliverable *Pilots evaluation framework, experimental plan and logistics V2*, [15]

The scope of this second iteration phase is to evaluate with real user or demonstrate and capture the opinion of the users for the Cloud4all prototypes that are ready to be tested. The prototypes to be tested at this iteration phase are either tested for first time, or they are updated and extended prototypes of tools that have already been tested during the first iteration phase. In this iteration phase, the plan is to go one step forward from the previous phase and try to test a scenario much closer to the final Cloud4all/GPII vision, including additional integrated components of the holistic approach of Cloud4all.

The auto-configuration scenario where this implementation will take part of can be summarized as follows:

1. The user sets his/her needs and preferences for using any application or device. This can be done using a web-based Preference Management Tool (PMT) [16] or snapshot the current settings of the current system.
2. The user can store a token in NFC tag or USB key.
3. The needs and preferences of the user are stored in the N&P server in a safe and secure way.
4. Whenever a user encounters a GPII-compatible device (a PC, a mobile device, an ATM, etc.), he/she can key in using the NFC card or the USB key where he/she has stored his/her personal token.
5. The device sends the token and info about its accessibility features to the Cloud4all/GPII infrastructure.
6. The architecture takes the needs and preferences of the user from the N&P server.
7. The matchmakers get information about the needs and preferences of the user, the device and the environment, and will calculate the most appropriate device settings for this user and this situation.
8. Now the user can use the new device without having to tweak any settings. If the device has not all the accessibility features needed, the GPII will recommend

solutions available in the Cloud. When the user keys back, the system will get back to its default setting.

People that will take part of this pilot iteration will test this auto-configuration scenario on many different devices and situations:

9. Using a PC operating system (either GNU/Linux or Microsoft Windows) with built-in ATs, platform specific features, and/or third party ATs.
10. Using a PC operating system (either GNU/Linux or Microsoft Windows) without any built-in ATs, platform specific features, or third party ATs. These will make use of cloud-based ATs or browser-specific extensions.
11. Using an Android device with the GPII running locally, making use of Android's built-in features and ATs, and third-party applications that are supported by the GPII.
12. Using an Android device without the GPII running locally, making use of third-party applications that make use of the GPII's Cloud-based Flow Manager.
13. Using a simple phone (JME-based phone platforms) with a third-party application that makes use of the GPII's Cloud-based Flow Manager.

Note that all the scenario can not be run in every device/situation but when applicable.

The following picture resumes very well the scenario and how the user will make use of the system:

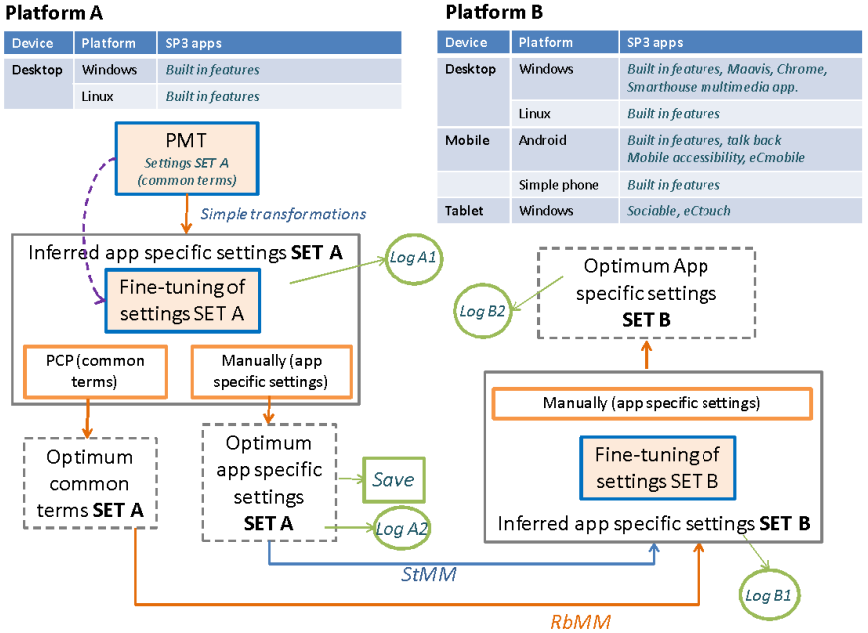

Fig. 2. Cloud4all second pilots scenario

In this scenario, the user will be asked to perform the following tasks:

14. To create an account (NP set). The user is asked to create an initial set of needs and preferences using the PMT, in a familiar desktop environment (Windows or Linux)
15. To edit inferred settings. The user is asked to edit/optimize the settings in this first platform, using the PCP [16].
16. To change platform. When the user keys in into a different platform, the auto-configured interface appears.
17. Edit inferred settings. The user is asked to edit/optimize the settings presented to his/her in the second platform.

With every step, a lot of sub-tasks will be performed by the facilitators of the pilots, and these will be the resulting data that will be used to evaluate the many different components of the system.

- The whole architecture itself.
- The PCP and the PMT as the available user interfaces to interact with the GPII.
- The implementations on many different platforms, applications, etc. ie:
  - GNU/Linux and the GNOME desktop, its built-in features, including the Orca screen reader.
  - Microsoft Windows, its built-in features, including the NVDA screen reader.
  - Maavis, as a standalone AT running on a locally running GPII windows platform.
  - Cloud-based AT solutions such as Read&Write Gold or Web Anywhere or browser-based AT solutions (such as Google Chrome's Cloud4Chrome), making use of the Cloud-based Flow Manager.
  - Android, its built-in features and the Talkback screen reader. Third-party applications such as Omnitor's ecMobile, by making use of a locally running instance of the GPII.
  - Third-party applications/ATs for Android such as CodeFactory's Mobile Accessibility For Android, which makes use of the Cloud-based Flow Manager.
  - Third-party applications/ATs in simple phones, such as CERTH's Cloud4allThemes application, which makes use of the Cloud-based Flow Manager.
- The matchmakers and its effectivity in inferring the user's need and preferences, and translating these from one platform into another and viceversa. In this phase two matchmakers will be tested:
  - Rule based Matchmaker
  - Statistical Matchmaker

All this data will be compiled, analysed and written into a deliverable as part of the Cloud4all project, as it will serve as feedback for all the people that are working on the GPII.

Acknowledgements. The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement 289016. The present work benefited form the input of the architecture team from the GPII project, and specially from Steven Githens, architect at the GPII, who provided the initial work of this implementation.

References

1. http://gpii.net
2. http://cloud4all.info
3. http://www.openhandsetalliance.com/
4. http://www.gartner.com/newsroom/id/2623415
5. http://online.wsj.com/news/articles/SB10001424127887323293704578334401534217878
6. http://www.afb.org/afbpress/pub.asp?DocID=aw130302
7. http://nodejs.org/
8. https://github.com/paddybyers/anode/wiki
9. https://github.com/GPII/android
10. http://wiki.gpii.net/index.php/Android_Overview
11. http://wiki.gpii.net/index.php/GPII_Android_FAQ
12. http://wiki.gpii.net/index.php/Common_Terms_Registry
13. https://github.com/duhrer/gpii-android-listener
14. https://github.com/javihernandez/android-user-listeners
15. Sainzet, F., et al.: Pilots evaluation framework, experimental plan and logistics, Cloud4all project deliverable D402.2.2
16. http://wiki.gpii.net/index.php/PCPs,_PMTs,_etc