Technological choices for mobile clinical applications

Frederic Ehrler, David Issom, Christian Lovis
University Hospitals of Geneva, Division of Medical Information Sciences, Geneva, Switzerland

Summary
The rise of cheaper and more powerful mobile devices makes them a new and attractive platform for clinical applications. The interaction paradigm and portability of the device facilitates bedside human-machine interactions. The greater accessibility to information and decision support anywhere in the hospital improves the efficiency and the safety of care processes. In this study we attempt to ascertain what are the most appropriate Operating System (OS) and Software Development Kit (SDK) to support the development of clinical applications on mobile devices. The Android platform is a Linux-based, open source platform that has many advantages. Two main SDKs are available on this platform: the native Android and the Adobe Flex SDK. Both have interesting features, but the latter has been preferred due its portability at comparable performance and ease of development.

Background
The Geneva University Hospitals (HUG) is a consortium federating the public hospitals in the Canton of Geneva, Switzerland. It provides primary, secondary, tertiary and outpatient care for the whole region with 45 000 inpatients and 850 000 outpatient visits a year [4]. The HUG’s Clinical Information System (CIS) is largely an inhouse-developed system. It is a service-oriented and component-based architecture with a message-based middleware. All exchanges are in SOAP or HTTP/XML [5, 6]. All component building blocks of the CIS, including those discussed in this paper, are built in such a way that they comply as far as possible with standards, such as IHE (Integrating the Healthcare Enterprise) profiles, so that they are not dependent on any local legacy system. This includes technical, semantic and human-machine interfaces, such as using a terminology server for the language of the interfaces.

Method
To define the most appropriate technology to develop mobile clinical applications, we defined several criteria organised along three axes:

- **Hardware**: market trends, cost, performance and user acceptance of the mobile devices. Strength of the mobile platform with regard to security, reliability and privacy.
- **Human**: availability of competent developers on the labour market and existence of a developer community.
- **Software**: complexity of the development environment, cost, user-friendliness and reusability of existing and new developments.

It is important to take into account the price of the physical devices supporting the OS, for when each care provider of the hospital is equipped with a mobile device, a small difference on price becomes really significant. The performance, including power autonomy of the device, is

Correspondence:

Frederic Ehrler
University Hospitals of Geneva
Division of Medical Information Sciences
Rue Gabrielle-Perret-Gentil 4
CH-1211 Geneva 14
frederic.ehrler@hcuge.ch

No conflict of interest relevant to this article was reported.
Comparison of the principal existing OSs to be developed on mobile devices

<table>
<thead>
<tr>
<th>Developer</th>
<th>iOS</th>
<th>Symbian</th>
<th>Android</th>
<th>RIM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language</strong></td>
<td>Objective-C</td>
<td>C++</td>
<td>Java + XML</td>
<td>Java</td>
</tr>
<tr>
<td><strong>Market shares</strong></td>
<td>46.4%</td>
<td>21.7%</td>
<td>15.6%</td>
<td>10.16%</td>
</tr>
</tbody>
</table>

Figure 1
Communication between mobile applications and existing CIS.

 obviously central, since a favourable course of the healing process often relies on real-time access to the relevant information. The information must obviously remain secured since it concerns the private life of the patient. In addition, we have to consider how quickly developers can master the environment and how easily the work already done inside the CIS can be adapted to the new tools. The choice of widely-used languages, such as ActionScript or Java, would definitely facilitate adoption and development since numerous developers are already familiar with these languages. The existence of a professional development environment, the existence of open source projects in this field, and an adequate developer community which has already addressed the most obvious questions, also facilitate development. To evaluate the features and the ease of development with the different SDKs, we defined a prototype mobile application, a sort of test use case, aiming to simplify the care process. With the help of this application, health professionals simply enter the information concerning the patient during the visit instead of recording all the information on laptops. The application is composed of a succession of screens where the user selects the unit, the room, and finally the patient currently examined. On the last screen the care provider can enter the vital signs of the selected patient.

Communication architecture

Regarding the architecture, it was mandatory to devise a model that would not create a dependency with any legacy system. Thus, we defined a gateway server providing centralised access for the mobile application to any required information or from the CIS. Thus, integrating any mobile application would only mean integrating this bridge. It also clearly separates the services that are available remotely from the ones proposed as usual Web services. The gateway server is responsible for formatting the data properly before sending it to the appropriate application on the device. Once the mobile device receives the data its embedded software is responsible for displaying the data through its interface and allows interaction with the user.

Figure 1 shows the link between our mobile application and the current CIS. The services of the existing CIS are externalised through a component named CIS gateway. When a mobile application requires data from the CIS it communicates with the mobile gateway that transmits the request to the CIS gateway. The service directory is then queried to identify the appropriate service from which to retrieve the required information. The information then returns through the same channel. All data transiting through the channel is formatted in XML.

Results

Choice of the OS

The choice of OS is a challenge. There are numerous OSs for mobile devices on the market, some of them with marginal shares. To simplify the work it was decided to address only the four that are currently seen as major players, as per the table 1.

The Apple iPhone is an interesting product since it is widespread among users [7]. Unfortunately, the development policy of Apple is very restrictive. In addition, the choice of devices, as only the devices provided by Apple are available on the market.

Choosing between Android, Symbian and RIM was trickier. They all possess a significant share on the market, rely on well established language and possess an efficient development environment. However, only Android offers a huge choice of devices, ranging from a very small Smartphone to large tablets, a widespread development environment, a large open source community, and a very transparent development policy.

Choice of the SDK

One would think it is straightforward to adopt the Android SDK to develop on the Android platform. However, it is worth taking into consideration Adobe, a major actor of the IT world that offers development tools for mobile devices running Android. Adobe provides an SDK named Adobe Flex that has the worthwhile advantage of generating programs that can be supported by several platforms unchanged. We rapidly surveyed (table 2) Adobe Flex and Android SDK characteristics to clarify their benefits and limitations. Some restrictions related to the Flex Hero SDK have been identified. As this SDK is an additional layer over the native SDK, there can be a loss of functionalities. Fortunately the Flex SDK can handle the main functions required to interact with the mobile device, such as positioning, multi-touch, inclination, etc. The only limitation identified is the impossibility of creating Android widgets, but this is not required for our application. The additional layer of the Flex SDK can also cause a reduction of performance. However, in our tests

Table 1
Comparison of the principal existing OSs to be developed on mobile devices (market shares of Western Europe, November 2010).
we did not observe and did not find objective and serious studies confirming or invalidating this fact. Regarding the Integrated Development Environment (IDE), the two languages possess a dedicated tool that helps developers generate accurate code. For Android SDK, Eclipse IDE is perfectly adapted as the code is standard Java language. With the addition of a plug-in, the Eclipse IDE can manage the installed SDK, the documentation, and some drivers to connect the mobile device to the computer. The plug-in offers automated compilation as well an emulator. It allows testing of the application locally instead of loading it into the mobile device.

For the Flex SDK, a new version of their development environment, Flex Builder, has been released recently by Adobe to program mobile applications. This IDE based on Eclipse offers programming facility to code in ActionScript and MXML. As with the Android SDK, there is an emulator that significantly facilitates the development.

Comparing Platforms

To improve our comparison we developed our sample application on the two platforms. In the figure 2 it can be seen that there are no marked differences in the human-machine interaction experience between the two interfaces. Both can display and manipulate lists, radio buttons and text inputs and other graphic component.

Conclusion

Our constraints, needs and projects led us to prefer the Android OS due its compatibility with the largest number of devices and its open source policy. The selection of the SDK was more difficult as both the Android SDK and the Flex SDK met most needs in terms of features for the development of a mobile application on Android OS. The Flex SDK was finally chosen on the basis of its portability to other platforms at comparable performance and ease of development.

References