Hybrid app-development using an existing webapplication

David Norrestam
davidnorrestam@gmail.com

Philip Burenstam Linder
philip.burenstam.linder@gmail.com

Month Day, 2015

Bachelor's thesis work carried out at Lund University.

Supervisors: Flavius Gruian, Flavius.gruian@cs.lth.se
Albin Svensson, albin.svensson@trialbee.com

Examiner: Flavius Gruian, flavius.gruian@cs.lth.se

Abstract

This document describes the Master's Thesis format for the theses carried out at the Department of Computer Science, Lund University.

Your abstract should capture, in English, the whole thesis with focus on the problem and solution in 150 words. It should be placed on a separate right-hand page, with an additional *1cm* margin on both left and right. Avoid acronyms, footnotes, and references in the abstract if possible.

Leave a *2cm* vertical space after the abstract and provide a few keywords relevant for your report. Use five to six words, of which at most two should be from the title.

Keywords: android, hybrid, application, app-development, webapplication, integrating, webview

Acknowledgements

If you want to thank people, do it here, on a separate right-hand page. Both the U.S. *acknowledgments* and the British *acknowledgements* spellings are acceptable.

Contents

1 Introduction **7**
- 1.1 Background . . . 7
- 1.2 Terminology . . . 8
- 1.3 Problem formulation . . . 8
  - 1.3.1 Preconditions . . . 9
- 1.4 Litterature study . . . 9
- 1.5 Contribution statement . . . 10

2 Approach **11**
- 2.1 Method . . . 11
  - 2.1.1 Demo application . . . 12
  - 2.1.2 Recent work . . . 12

3 Evaluation **15**
- 3.1 Native android . . . 15
  - 3.1.1 Modularisation . . . 15
  - 3.1.2 Function calls between Java and JavaScript . . . 17
- 3.2 Web application . . . 17
  - 3.2.1 Results . . . 17
- 3.3 PhoneGap . . . 19
  - 3.3.1 Communicatating between PhoneGap (native) layer and the web application . . . 19
  - 3.3.2 Demo . . . 21

4 Discussion **23**
- 4.1 Developing in Android . . . 23
  - 4.1.1 Getting started . . . 23
  - 4.1.2 Debugging . . . 23
- 4.2 Developing in PhoneGap . . . 23

4.2.1 Getting started 23
4.2.2 Adding and debbuging native functionality 24

5 Conclusions **27**

Chapter 1

Introduction

Imagine a smaller company with a well developed web application. The company has a few developers and has the want to extend their web applications features for their users. They want to track their users activity for making a health profile. To track the user's activity they need to use the mobile accelerometer. The accelerometer can be accessed through a mobile application but not through a web application.

The company can choose to develop a mobile application from the ground up and integrate the application with their current web application. That will cost them a huge amount of development time and also a lot of time maintaining several platforms.

Another less costly alternative is to create the mobile application using the existing web application. The web application is run within a shell of the mobile application. The web application can then access the mobile's native functions such as the accelerometer. This will save the company time developing the mobile application and instead of maintaining several platforms for every feature the mobile application only need to be maintained in regards of the use of the accelerometer.

The aim of this thesis is to evaluate two different development methods for encapsulating an existing web application in a mobile application to utilize native functions. With focus on evaluating development and maintenance costs.

1.1 Background

Android is an operating system used on a wide range of devices. For example a mobile device or tv device. Developing Android applications are written in the

programming language Java.

A hybrid application is a native application which is partially written with web technologies. I.e HTML5, CSS and JavaScript. The part of the application written with web technologies runs within a browsers engine in a so called WebView. A web application running in a browser does not have access to a device native functions such as the bluetooth. However a website running within a WebView can through interaction with the application's native code access a device native functions.

PhoneGap is a framework for developing mobile applications. The code is written with web technologies. The code can be compiled to different platforms, such as Android or IOs specific code. The resulting application is a hybrid application. An example of a mobile application built in PhoneGap is Wikipedia's mobile application.

1.2 Terminology

Mobile application
: Refers to the application being developed, excluding code loaded from remote websites.

Web application
: If nothing else is specified it refers to the client-side of the web application.

Development methods
: Refers (in the context of this paper) to the methods of mobile application development methods being evaluated. I.e developing the mobile application with PhoneGap or natively in Android.

Native function
: A hardware function which a device has. Lets say a mobile has a camera and an accelerator. When writing software for that mobile there are functions to interact with the camera and accelerator. Those functions are called native functions.

1.3 Problem formulation

The goal of this project is to evaluate two different development methods for encapsulating an existing web application in an Android mobile application. The purpose of the mobile application is to give the web application access to the mobile's native functions. An example of such a native function would be accessing the mobile's accelerometer. It is important that the logic of the web application can be written in a general way (non-platform dependant) adapting to the device accessing the web application. So that the same web application code is used for the web application and mobile application.

Antoher goal is that the mobile application only provides data from native functions. In order to achieve that the web application and mobile application should have a master slave relationship. Where the web application acts as the master and the mobile application as the slave. To get data from the mobiles native functions the web application asks for the data and the mobile application passes the data back.

To limit the scope of the research the development methods is only evaluted for the mobile operating system Android. Android was choosen since it is the most widely used today for mobiles.

There are two development methods of creating an Android application that will be evaluated. The first development method, the mobile application will be built natively in Android. As the second method, the mobile development framework PhoneGap will be used. The methods will be evaluated and compared from a developers perspective, with the preconditions defined section 1.3.1.

1.3.1 Preconditions

The study is done from a company's perspective and is based on the following prerequisites:

- There are only a few developers, three or less.
- There is already an existing web-application that is to be used by the mobile-application.
- The company has a need and/or desire to extend the functionality of it's web-application with native functions.

When examining the methods, the focus has been restrained to the following developing aspects:

- Development time.
- Maintainability when the application has been created.
- Programming productivity when the application has been created.

1.4 Litterature study

Source lines of code is a measurement tool for software development. Source lines of code, also abbreviated as SLOC, is very easy to obtain and is a fairly accurate predictor of development effort[1, p. 63]. There are many different ways to measure SLOC, such as Halsted's approach, function points, physical SLOC and Logical SLOC.

Physical SLOC is the length of the code excluding comments and blanks. Function points measure functionality and can therefore be measured before the design and

coding if the requirement specification complete[1, p. 187]. Halsted's uses measurable properties such as operands and operators and uses them to identify properties of software. Such as the length of the program, difficulty and effort[2, p. 345]. Fenton and Bieman describes Halstead's software science measures as a confused and inadequate measurement. Particularly for other attributes then size.

Logical SLOC measures the number of statements that carry over one or more several physical lines. For languages with terminators this can be done more easily and quickly. As an example in Java you could count the logical SLOC by counting the number of line-terminating semicolons and closing curly brackets. Logical SLOC represents the programming instructions and data declarations which are converted into executable instructions i.e. the implementation of the software design. Another positive aspect of of logical SLOC is that it better handles differences in formatting and style conventions than physical SLOC[1, p. 155].

To compare size between two different languages a size conversion table can be used. The table can be used to estimate how many SLOC a program coded one programming language would have to be written in another language. You can compare a third generation language, a fourth generation language, Ada, Assembly or Pascal. A third generation language compared to another third generation language would have no conversion rate[1, p. 163].

1.5 Contribution statement

Throughout this work we have been working together very closely. We have reviewed and improved each others work continuously. During development we often been pair programming. With that said we can attribute parts of our work to one of us more then the other. Even though most of the work has been created together.

Philip designed the code of the web application and the mobile application built natively in Android. David designed the code for the PhoneGap application. David researched different techniques for the communication between the PhoneGap layer and the web application layer. David wrote the Approach section and Philip wrote the Introduction section in this paper.

Chapter 2

Approach

This chapter begins with a presentation of the method used in this thesis, followed by a section explaining the demo application developed as part of the method. The last section contains a presentation of recent articles on the subject.

2.1 Method

?? In order to compare the two development methods (described in problem formulation), we have tried using them both to

In order to research the problem formulation of this paper, a practical approach has been used. The two different development methods (as described in problem formulation) have been used in a practical development process. The aim of the development has been to develop an architecture allowing independent web application code and mobile-application code to be written.

Apart from just developing architectures using the different methods, a demo-application was developed. The application had the same function requirements for both development methods.

This also put requirements on the web application, it had to confirm to a certain standard for communication with the mobile application in order for both applications to be able to use it.

Another important aspect of the method, and a major part of the results found in this paper is the process of researching questions arising during the process. Examples of this can be questions regarding strengths and weaknesses of the method

at hand, but also questions of a more basic nature, such as the behavior of function calls between the layers (web and mobile), are they synchronous?

2.1.1 Demo application

The demo application consists two mobile applications developed using the different development methods. The web application which is used in the mobile applicaitons is a simple form with a text field, an image field and a location field.

When running on a computer in a web browser the web application make use of the computer's camera and the web browser's own function to get the computers location.

The two mobile applications make use of the web application described above. To get the image for the web application form the mobile application makes use of the mobile's native camera function or uploads an image from the mobile's memory storage. To get the location the mobile applications makes use of the mobile's native GPS function.

Application source code can be found at:

- Web application: `https://github.com/Albin-trialbee/pollux-server`
- PhoneGap application: `https://github.com/DavidNorrestam/pollux-phonegap`[redacted]
- Native Android mobile application: `https://github.com/buren-trialbee/pollux/`

2.1.2 Recent work

In "Native vs Hybrid / PhoneGap App Development Comparsion" (`http://www.comentum.com/phonegap-vs-native-app-development.html`), research made by Comentum 360, they estimate the development cost for a small to medium size project in Android native development and in PhoneGap. They estimate that both methods would cost the same, about 10-20 thousand dollars. They also mention the limitations of using PhoneGap, describing that hybrid apps (like PhoneGap) relies on a development framework and the provided features when building a mobile application. Hence, if the framework is not up to date with the latest new features, the developer will not be able to partake of the features until the framework is updated. Their conclusion of this is that if an application is dependent on many native features a hybrid application may have limitations.

TJ VanToll author of the book "JQuery UI in Action" writes amongst other things in the article, "The State of Hybrid Mobile Development" (`http://developer.telerik.com/featured/the-state-of-hybrid-mobile-development/`), about the[redacted] history of hybrid mobile development using web technologies. He points out that there has been a lot of issues such as debugging and lack of tools. Two things which

made companies like Facebook and LinkedIn abandon developing native applications with web technologies. Troubles, he today believes mostly are solved with all the progress which has been made the recent years.

Chapter 3

Evaluation

The evaluation has been divided into two different parts, one for each development method being evaluated. In

3.1 Native android

3.1.1 Modularisation

The android application has a lot of different responsibilities, apart from communicating with the native API and web application, it also has to contain the necessary activity threads to function as an application. A modularisation of the code into as small sections as possible has been the goal of the architecture being developed. The different responsibilities can be summarized as:

- Communication with the Android native API
- Communication with the web application
- Handling activity lifecycle
- Starting activities for results (ex. the camera)

To conform to this modularisation, the architecture as figure 3.1 has been proposed. This divides the application into separate modules with a single responsibility, where the bridges responsibility is to handle communication between modules and add necessary logic for simple tasks such as casting of objects or performing calculations.

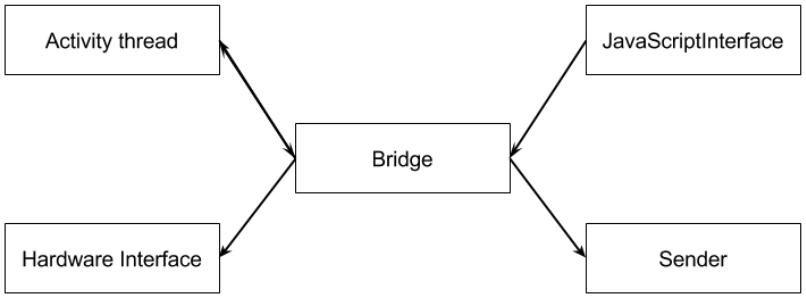

Figure 3.1: Architecture of the android application

(Add section describing function flow between java and javascript)

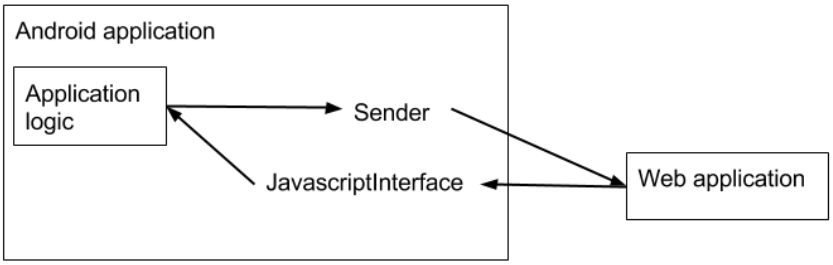

Figure 3.2: Function flow of the android application

3.1.2 Function calls between Java and JavaScript

Calling javascript functions from Java is done using the function loadUrl on the WebView containing the web application. This was deducted to be an asynchronous method call, through practical tests in the development phase of this paper.

(Add section containing example test here)

When calling Java functions on the exposed javaScriptInterface object from JavaScript, the behavior of the function call is a bit more obscure. Due to the single-threaded nature of javascript, function calls are synchronous when called in a normal fashion.

(Add section containing example test here)

(However, by the usage of AJAX, the function calls may behave asynchronous on the web application side. Then the question arises, can two functions be executed in parallel on the android side, or is the execution of the jsInterface functions limited to being handled by a single thread.)<–Needs to be researched to stay here.

Utvärdering (Evaluation) bör beskriva den metod som används för att utvärdera din metod, inklusive experimentuppställning.

3.2 Web application

3.2.1 Results

Separating business logic from device specific logic

An requirement on the web application is the ability to separate the business logic from device specific logic. This can be done with an adapter pattern. To set the right adapter a different method is used for Android and PhoneGap. The usage of this pattern ensures a natural way of modularising the code and separating all business logic from logic related to device-specific communication, the business logic is thus independent of the running device.

Encapsulating the web application in native Android application

When encapsulating the web application when writing the mobile application in native Android code is done by the use of a WebView. When the web application is loaded in the native layer through a WebView an object is made available to the web application on which the web application can use to send messages to the native layer. Therefore checking if the web application is running within a WebView can be done by:

```
if(typeof Android !== 'undefined'){
 adapter = new AndroidAdapter()
}
```

Where "Android" is the object made available to the web application by the native layer and can be used to invoke native functions.

Encapsulating the web application in a PhoneGap application

Here the same type of text as the chapter above shall be written

An overview of the recommended architecture for the web application can be seen below, where the internal structure of the mobile application is irrelevant.

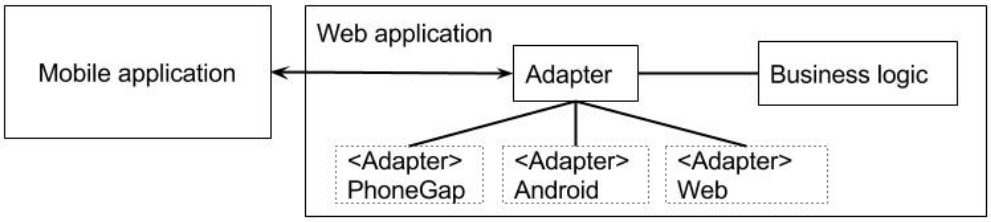

Figure 3.3: Web application function calling flow

Communication between web application and mobile application

(this chapter shall be rewritten) Another functionality raising concerns regarding dependency is the communication between the web application and mobile application. The adapters dependency on the mobile application code is inevitable, unless communication can be made in a standardized way, and all devices expose methods conforming to an interface. The same thing goes for communication back to the

web application, however in this case, the web application can expose a javascript function handling device callbacks, and as long as all devices are able to invoke javascripts in the web application, communication is independent apart from the name of the callback function. Ok that was not entirely true, in order for communication to be independent, all calls to the device must include the name of a callback function that will later be passed to the device callback function in the web application, which then relays callback data correctly. Also, the dependency on standardized callback data is still there, however this is a wanted behavior, as the data being sent back should be of the same type.

3.3 PhoneGap

3.3.1 Communicatating between PhoneGap (native) layer and the web application

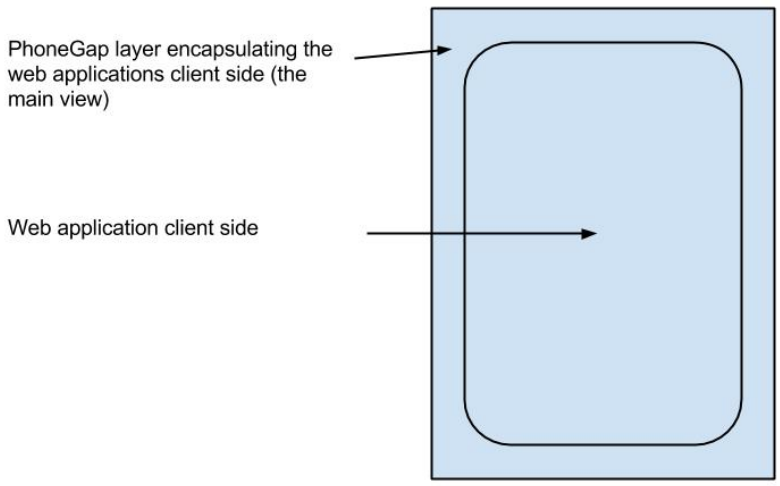

Figure 3.4: The PhoneGap layer encapsulating the web application

Encupsulating the web application in an iFrame

The remote web application is loaded in an iFrame, whilst PhoneGap logic is running in the main view (see picture). The default response header for a web page contains rules regarding same-origin policy, which disallows the webpage from being

embedded in an iFrame on another web page, unless they have the same origin. The response header therefore needs to be modified, in order to allow web pages with other origins to load the webpage in an iFrame. This is done differently depending on the web development framework, in the framework Rails this can be done by:

```
HomeController  < ApplicationController
  after_action :allow_iframe
  private
    def allow_iframe
      response.headers.except! 'X-Frame-Options'
     end
```

Function calling from the web application is done by the use of postMessage on the parent window, the PhoneGap-layer listens to these messages by the use of an event listener. A request call from the web application is a message containing two key-value pairs: "type" and "callback". The value of "type" specifies which native function the web application desires data from. The value of "callback" is used by the PhoneGap-layer when sending the data back to the web application. This is done with the predefined method deviceCallback on the adapter in the web application:

```
Adapter.deviceCallback(nativeData, calbackName);
```

This way the PhoneGap layer only need to know in which format the data should be in, nothing else.

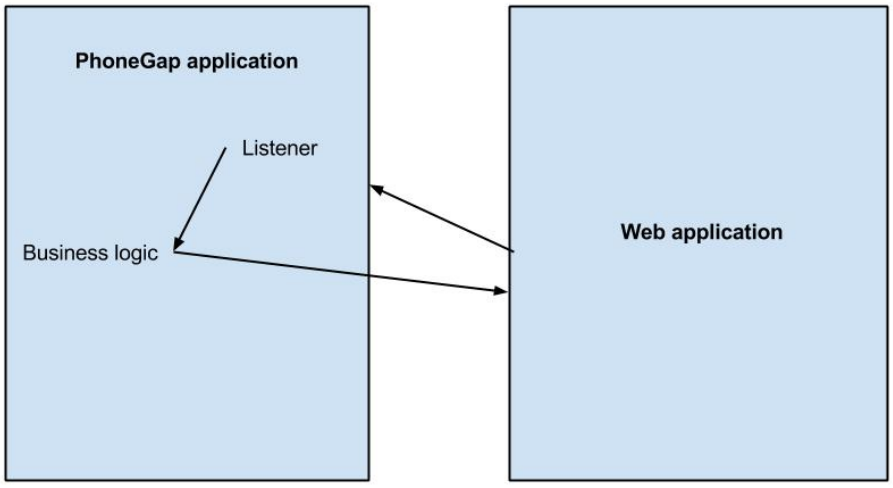

Figure 3.5: PhoneGap function calling flow

The web application sends a message which is received by the listener in the PhoneGap application. The listener calls the function specified by the message, and passes

on the callback to it. Upon completion the resulting data is passed to the web application, along with the callback,

Using polling in the PhoneGap plugin inAppBrowser

inAppBrowser is a PhoneGap plugin which basically opens a browser "in the application". By using this plugin to embed the remote web application, communication between the windows becomes different than when using an iFrame. Sending data from the PhoneGap-layer to the web application can be done by using the inAppBrowser function "executeScript", which, in the inAppBrowser, executes the javascript passed as argument.

The communication back from the web application is more complicated. One example of how to do it is by the use of polling variables in the web application using executeScript. For further reading the following blog article: "Cross Window Communication With Cordova's inAppBrowser" (`http://blogs.telerik.com/appbuilder/posts/13-12-23/cross-window-communication-with-cordova%27s-inappbrowser`) by TJ VanToll is recommended.

Other ways requires deeper technical web development knowledge. For more suggested methods read the support request https://issues.apache.org/jira/browse/CB-4897.

Using a third party plugin

PhoneGap supports custom plugins. This opens up the ability for companies to write their own plugins or make use of other third party plugins.

There is a PhoneGap plugin written by the corporation Wizcorp which claims it enables communication between the PhoneGap layer and web application. However despite extensive effort it never worked in this project. A ananswered support question (25/3-2015) can be seen at https://github.com/Wizcorp/phonegap-plugin-wizViewManager/issues/77.

3.3.2 Demo

The demo code can be found at `https://github.com/DavidNorrestam/pollux-phonegap`. In the demo the communication between the PhoneGap layer and the web application is done by using an iFrame and postMessage, see chapter above. The most interesting files in the repository are found in:

www/js/receiver.js
: All the logic for receiving a message from the web application

www/js/device.js
: Logic for some native functions and loading the web application

www/index.html
: The html code for the mobile application view

Chapter 4

Discussion

4.1 Developing in Android

4.1.1 Getting started

4.1.2 Debugging

Debugging when using the Android Studio IDE is a fairly simple process. For developers familiar with Java-debugging, the process is as good as similar, apart from the fact that the code is run on an emulator, or real device, rather than on directly by the JVM. Complications do occur when the application contains a WebView or other browser and the aim is the debug both the web and java code. However, with the help of the chrome inspect tool, the web code is easily debugged, and debugging of communication in between Java and JavaScript is easiest made by a combination of this tool and the built-in debugger.

4.2 Developing in PhoneGap

4.2.1 Getting started

Getting familliar with PhoneGap is quite confusing. This resulting in that it took around two days to set up a working development environment that felt comfortable.

There were a few reasons for this. You need to install a lot dependencies such as nodejs, a server-side runtime environment, and Apache Cordova. Apache Cordova can be described as the engine of PhoneGap. There is also several different ways to develop the mobile application. You can compile and build the app for testing yourself or use a cloud service like https://build.phonegap.com/ to compile your code for you. The relationship between Apache Cordova and PhoneGap is also initially confusing. It was also very difficult to find a good "Getting started" guide.

4.2.2 Adding and debbuging native functionality

The documenation is clear and feels complete. If you require extra help the PhoneGap community is very active and there is lots of help to get from blog posts and Q&A sites such as stackoverflow.com. This make's it very simple to make use of the mobile's native functions.

When using mobile native function a plugin need to be added to the project. The plugin can be supported by PhoneGap, a third party plugin or a plugin you write yourself. A PhoneGap plugin for using the mobile's cambera is added by writing the following bash command in your project root folder:

```
$ cordova plugin add org.apache.cordova.camera
```

The code for taking a picture with the camera in your mobile application is:

```
navigator.camera.getPicture(cameraSuccess, cameraError, cameraOptions);
```

Where "cameraSucess" and "cameraError" is a callback function.

To build and run the project on your mobile for testing you run in the root folder of your project:

```
$ cordova build android;
$ cordova run android;
```

For the debbuging the mobile application you can open desktop browser and where you can see the HTML, CSS and JavScript code and debugg the same way as a web application debugging is done in a web browser inspector. More can be read about debugging a PhoneGap application at `https://github.com/phonegap/phonegap/wiki/Debugging-in-PhoneGap`.

Thoughts: - The development of the architectures of the different methods can not be ensured to be entirely independent. Thus the results might vary depending on the order the development methods get evaluated. Also they're both reliant on the webapplication, if this one gets developed early on, they will have to conform to its standards, thus limiting the developers freedom.

Positive / negative experiences

- We are both skilled in java
- Very basic knowledge of webprogramming

- Our experience of developing in android
- Our experience of developing using phonegap
- Other

Diskussion (Discussion) möjliggör en längre diskussion och tolkning av resultaten från utvärderingsavsnittet, inklusive extrapoleringar och/eller förväntade resultat. Här passar det också att beskriva positiva och negativa erfarenheter relaterade till det arbete du har utfört.

Chapter 5

Conclusions

Slutsatser (Conclusions) bör sammanfatta dina slutsatser och redogöra för möjliga förbättringar och eventuella rekommendationer.

Bibliography

[1] Daniel D. Galorath and Michael W. Evans, *Software Sizing, Estimation, and Risk Management*, Auerbach Publications, Taylor and Francis Group, 2006.

[2] Norman Fenton, Queen Mary University of London, UK, James Bieman, Colorado State University, Fort Collins, USA, *Software Metrics: A Rigorous and Practical Approach*, CRC Press, Taylor and Francis Group, 3rd edition, 2015

[3] Norman Fenton and James Bieman, *LaTeX: a document preparation system*, Addison Wesley, Massachusetts, 2nd edition, 1994