On the Experience of Using Git-Hub in the Context of an Academic Course for the Development of Apps for Smart Devices

Rita Francese,[1] Carmine Gravino,[1] Michele Risi,[1], Giuseppe Scanniello[2] and Genoveffa Tortora,[1]

***Abstract*— In this paper, we present the experience we gained in a Mobile Application Development course for Computer Science students at the University of Salerno. The course foresaw a project work conducted by students organized in teams. The goal of the project work was to design and develop Android-based applications for smart devices. The learning approach was based on collaboration (intra-team) and competition (extra-team). Students cooperated using GitHub as Computer-Supported-Collaborative-Learning tool for the implicit and explicit communication among team members and distributed revision control and management of software artifacts (e.g., source code and requirements models). All the developed applications underwent a final public competition prized by IT managers of national and international companies. IT managers expressed a positive judgment both on the students' competition and on the developed applications for smart devices. Also, the students provided very good feedback on the competition and on the GitHub support.**

I. INTRODUCTION

Internet and mobile applications are converging and from their union a new society is going to appear. According to Gartner [1], the interest of business users and customers in mobile devices and applications is increasingly growing. The digital enterprise becomes a mobile enterprise. In addition, mobile devices offer a rich set of embedded sensors, such as accelerometer, digital compass, gyroscope, GPS, microphone and camera. These sensors enable the production of new applications addressed to a wide variety of domains. In this scenario, the mobile application developer is one of the most demanded and fastest growing IT career.[1] Nevertheless, the development of mobile applications is not an easy task: the developer is required to master a wide range of technologies and capabilities, including programming languages (e.g., Objective C, C++, C# or Java) and operating systems (e.g., Android and iOS) [2], [3], [4], [5].

In this paper, we present the learning experience related to the second edition of the mobile application development course that fosters teamwork and encourages students to explore new ideas. Indeed, students were required to design their applications for smart devices (also simple apps, from here on) by considering the market needs, usefulness, audience, and viability. The course was organized in blended learning modality: the lectures on the Android operating system were given in presence, while students' projects where asynchronously managed by using GitHub, a largely adopted tool in technology areas that require collaboration and, more recently, also in education [6]. During the analysis and development activities the students exploited the distributed revision control, source code management (SCM), and asynchronous communication functionalities offered by GitHub. The communication among students took place also implicitly[2] through the developed software artifacts (e.g., software models). The GitHub use also allowed us to support a learning approach based on collaboration (intra-team) and competition (extra-team). The lecturer and two tutors supervised the projects by fixing strict deadline and monitoring the project status on GitHub. A distinguished panel of corporate IT managers were asked to judge and give a prize to the three best apps produced during this course. The selection was based on the team live presentations conducted during a public event organized at the University of Salerno. Indeed, IT managers judged for each app its originality, the estimated business value, the pleasantness of the User Interface, the estimated technical quality, and the team presentation. To complete our study, we also performed a qualitative evaluation on the student's opinion concerning their learning experience and the used technologies.

The paper is structured as follows. In Section II, we discuss background. In particular, we provide the concept of Project-Based Learning and the technological solutions adopted for it, and successively we describe the main issues behind the mobile development in Android. In Section III, we detail our experience, while we describe the evaluation performed by the industrial partners and the collected student perceptions in Section IV. Final remarks and future work conclude the paper in Section VI.

II. BACKGROUND

A. *Project-Based Learning*

Project-based learning (PBL) is a model that organizes learning around projects [7]. It is based on both the constructive learning theory [8], where learners become active constructors of their knowledge, and on cooperative/collaborative learning [9], [10]. PBL enables students to cooperate in solving real problems, performing activities typical of the job world, which results in higher student involvement. The production of an artifact that is of interest since others can use or view it is a very motivating factor. Motivation can make the difference between success and failure of a learning experience more than any other factor.

[1]University of Salerno, Italy {francese, gravino, mrisi, tortora}@unisa.it
[2]University of Basilicata {giuseppe.scanniello}@unibas.it

[1]http://www.itcareerfinder.com/it-careers/mobile-application-developer.html

[2]It is a knowledge transfer process based on communication through a shared mental or abstract model.

DOI reference number: 10.18293/DMS2015-003

The instructor has a less central role, and students are responsible for their own learning (learner-centered education [11]), while learning is generated by their interaction (learner-learner interaction) [12]. When this methodology is supported by technology it is empowered by the capability of engaging learners by providing rapid, compelling interaction and feedback.

The adoption of PBL in Computer Science courses is growing [13], [14]. The main reason is that it enables to train students in principles, methods and procedures under conditions similar to developing real software products [15]. The development of a software product is the result of the team effort which requires both technical skills and soft skills, including the ability to communicate, to work as a team, to partition, assign and monitor task progress, and to assume responsibility for making choices. In the various project phases there is also the need of producing documentation which follows determined standards and not only to concentrate on the coding activity [14].

B. Technology Support for PBL

Several technological solutions have been proposed to support PBL in computer science courses [14], [16], [17]. As an example, Macias [17] adopted a Moodle-based e-portfolio to support PBL activities involving a lot of deliverables and organizational resources.

Ardaiz-Villanueva *et al.* validate the effectiveness of Wikideas and Creativity Connector tools to stimulate the generation of ideas and originality by university students involved in PBL activities [18]. On the other hand, Zagalsky *et al*. [6] examined how GitHub is adopted as a collaborative platform for education. GitHub initially supported code and project management for software development; recently it is used also in other domains that involve collaborative work, including education, mainly for managing students and their work. It is exploited as submission platform, for hosting course material.

Kizaki *et al.* use GitHub as supporting tool for an Agile course consisting in scrum-based PBL [14]. The paper is mainly focused on the scrum methodology.

In this paper we conducted an in-depth case study of how GitHub plays a role in a specific course related to the learning of an emerging technology, Android-based apps development.

C. Mobile application development in Android

One of the mobile app challenges is to deal with multiple platforms during mobile development [13]. Developers can create mobile apps by using either native development tools for each of the major mobile platforms, such as iOS, Android, Microsoft Windows Mobile, Symbian, BlackBerry, or cross-platform environments, including PhoneGap and Titanium [2], [4]. At the present, developers separately create the app for each platform. Indeed, the features of a specific operating system may not be available in another. Alternatively, developers can develop a cross-platform app that runs on any environment, but has more limited functionalities. For example, to create an app that exploits in the better way the features of an Android device, developers have to master development skills related to the Android operating system and the associated development environment and resources. Android is an operating system whose demand is immensely expanding day by day.

As for the available resources, smartphones are equipped by sensors, such as accelerometer, gyroscope, GPS, brightness and temperature, offer communication features, including phone calls, SMS, email, and camera functionalities. The main Android components are: the activity and the service.

An activity is an app component that provides a screen with which users can interact in order to do something, such as dial the phone, take a photo, send an email, or view a map. Each activity is given a window in which to draw its user interface. For simplicity reason, we will refer to an activity as a GUI.

The app execution flow is continuously interrupted by the verification of asynchronous events. For this reason, the developer has to implement the activity logic taking into account its life-cycle. For example, when an activity is suspended (e.g., for the arrival of a phone call) the app has to perform specific work that is appropriate to that state change, or, when the device is rotated, an appropriate GUI has to be shown.

A service is a background component that performs either long-running operations or works for remote processes. A service does not provide a user interface. For example, a service might fetch data over the network without blocking the interaction the user has with an activity.

The market of Android hardware devices is very fragmented in terms of different screen sizes, processor types, custom APIs, etc. The main challenge is to maintain similar execution performances and user experiences in all these variations. Also testing is difficult, since it is practically impossible to test the app on all the available devices and OS versions. Android manages different configurations by exploiting non-code app resources (images, strings, layout files, etc.), which should include alternatives for each considered configuration.

Developers can create the GUI of an Android activity directly in Java or by using an XML-based layout file. The latter approach has two main advantages. It allows to 1) separate logic from presentation; 2) to maintain different parallel layouts for difference screen sizes.

The adoption of XML for specifying GUI requires, rather than setting the content view to be a view created in Java code, setting it to a reference to the XML layout.

III. Our Teaching Experience

Goal. The main objective of the Mobile Application Development course was to increase student interest, knowledge and practical experience in mobile development through an engaging and empowering PBL experience.

The software platform. The course was focused on the Android operating system because the barriers to entry in Android remain much lower. Indeed, with respect to iOS,

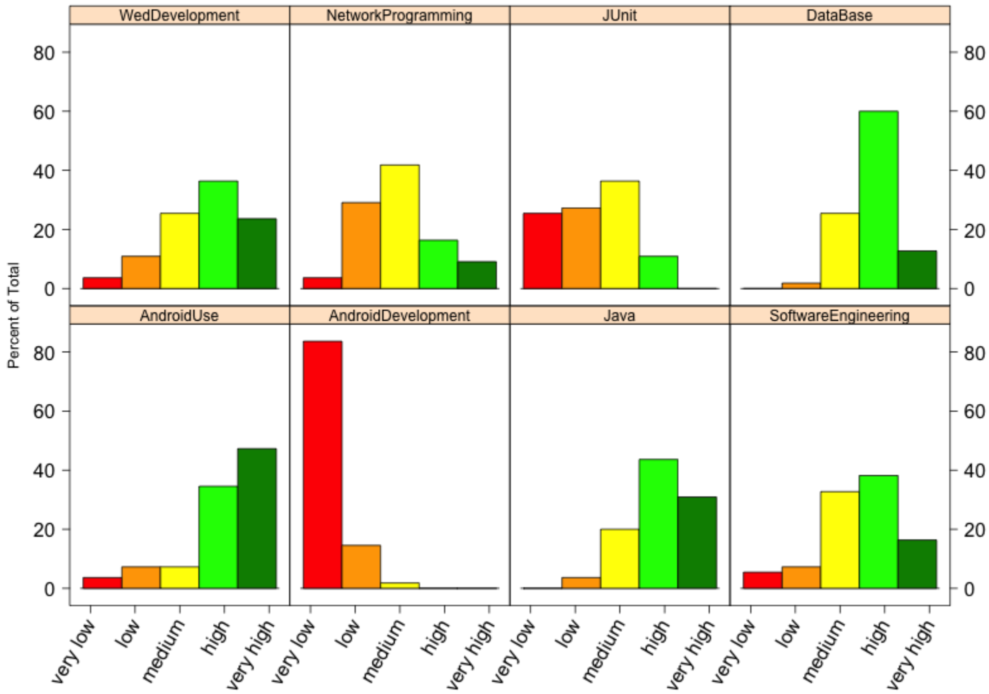

Fig. 1. Students' development competencies.

developers can iterate and test their designs quicker on Android, and marketing costs are significantly lower. At the present, Android users grow and are the largest overall smartphone market [1].

Context. The students were 55 Computer Science students at the University of Salerno. They were enrolled to the third year of the Bachelor program. Before the course, the students were asked to fill in a pre-questionnaire, which included questions (i.e., statements) regarding the following points: experience on the Android device use (as smart-users) and development, knowledge on Software Engineering and Web Development principles, experience on Java and Network Programming, and JUnit and Database knowledge. The questions admitted answers according to a 5-point Likert scale. Possible answers ranged from "Very Low" (1) to "Very High" (5). As shown in the histograms of Figure 1, students are generally smart-users, they did not know Android Programming, most of them thought to be good Java developer, 30 students affirmed to have a good competence in Software Engineering, 33 to be good web developer. Less (14 students) declared to have good network programming abilities. This was due to the contemporaneity of the network programming and the mobile development course. Very few known JUnit, while most of them were experienced in database development.

Course organization. The course consisted in two main modules: *i)* lectures on the Android operating system; *ii)* project work concerning the development of an app for smart devices accessing services available on the web. The course lasted 12 weeks. The course was performed in blended modality. In particular, the first part of this course was in presence, with the didactic material available on-line on our learning platform. The course topics were the following: Android Activity lifecycle, modern interfaces, accessing to the web, threads, Android services, access to RESTful web service by JSON, accessing to native functionalities (i.e., GPS, sensors, camera, SMS, call), monetization. The second part of the course was conducted in distance modality and consisted in a project work, as better detailed in the following of this paper.

A. Project organization

Teams. Students were divided in 27 teams of two people, except one of three. The pedagogy of project-based learning suggest that to obtain good results groups should be composed of students with similar ability and interest in the topics being learned [19]. However, the debate on the effectiveness of homogeneous and heterogeneous groups is still open and needs further investigation [20], [21]. Thus, teams were composed according the students' preferences. We did not decide to randomly assign team members because the students had previous experience of project work in several courses and, at the last term, they know which is the classmate more appropriate to work with.

Software Projects. Each team directly proposed the app to be designed and developed in the project. The rationale behind this choice was related to stimulate student's entrepreneurship actions and creativity. Each proposal was accurately motivated, by performing a detailed market analysis. The projects had to respect the following nonfunctional requirements: the app had to interact with a remote server, through JSON. It had to exploit native device functionalities, including maps, GPS, sensors, call, SMS. It had to handle device rotation and to use SQLite. Games were admitted if they exposed backend functionalities, such as account management, multiuser, bonus management, and app upgrade. The project started after the approval of the lecturer.

Lifecycle model. We addressed the students to follow an incremental prototyping development lifecycle. We did not discourage the adoption of pair programming, namely a software development practice where two programmers work as a pair on the same computer. Pair programming is an effective practice largely adopted in industrial settings [22]. As for students, it has been observed that their performances improve [23], namely they produce higher quality source code, are more confident in their work, and enjoy this more.

Deadlines. Students were informed of the project deadlines related to the presentation of the deliverables. The first deliverable was the project proposal, the second was the Requirement Analysis Document (RAD). Successive deliverables are referred to the GUI prototype, the mobile app prototype, the complete app, including the external server.

Documentation. The documentation required had to follow the templates proposed by the lecturer. In particular, a Project Proposal has to present the idea underlying the project, the motivation, a summary study of the market, also considering the apps available on the main app stores, and has to convince about the novelty of the proposed app. The RAD had to better detail requirements and also provide system models (actors, use case diagrams, class diagrams, sequence diagrams, navigational diagrams and user interface Mocks up). Black box test cases had also to be produced in order to test the final version of the app.

B. On the Use of GitHub

Student projects were managed through GitHub, a distributed revision control and source code management (SCM) system [24]. It currently hosts over one million code repositories, and has 340,000 registered contributors. Each repository on GitHub has a dedicated project page that hosts the source code files, commit history, open issues, and other data associated with the project [25]. As investigated by Zagalsky *et al.* [6], GitHub can be a powerful learning management tool, differently used by various educators even in similar environments (e.g., technical background) and with similar requirements (e.g., class size, course type).

The lecturer and the tutors monitored the quality and the times of the projects, supervising that teams correctly performed their work to be able to participate to the public App Challenge. The lecturer creates a GitHub account for each team, downloading on it the documentation templates to be provided. In particular, he uses the GitHub mechanism for milestones, typical of many project systems. A new milestone simply has a title, description, and a date. GitHub also provides a graph view that summarizes project activities. In this way the lecturer had a high-level view of the students' activities during the app development.

The team uses the GitHub communication feature. In this way, the communication is handled in one centralized place rather than across emails and visible to all the team. Using labels team members (also the lecturer) can create issues for discussion. Team members can set up email notifications when people comment or tag them in an issue. The communication features offered by GitHub favor awareness, which has a very relevant value of activity information for small teams [26]. Indeed, notifying members of actions on shared artifacts helps them maintain mental models of others activities [27] and avoid potential coordination conflicts [28]. In particular, when a deliverable was completed, the team notified the lecturer that it was ready. The lecturer could accept it or notify the change to perform by adding a checklist-based revision to the teams' GitHub account. The communication between the lecturer and the team members was easier with respect to other learning management systems.

It is also important to point out that the transparency on GitHub supported learning from the actions of other students. Indeed, they are able to look at the documentation of students of the same team and of students of other teams, how the other students coded, what they paid attention to, and how they solved problems. The availability of this information enables them to learn better ways to code and access to superior knowledge [25]. Also competition is favored, since a team can monitor the state of the others and is stimulated to perform better.

IV. Evaluating Team Work and Developed Apps

The evaluation of the team work and the developed apps went through the following evaluation levels:

- *Lecturer.* The artifacts each team produced were constantly monitored by the lecturer. In particular, she used GitHub to monitor the progress of the projects and to assess whether teams respected deadlines for the delivery of software artifacts. GitHub was also used to enable the communication among the lecturer and the students. The communication among the lecturer and the students took place in presence when needed. For example, the students had to show three versions of their app and in this case revision meetings were planed and conducted in presence. A wrap-up meeting was also conducted before the App Challenge[3], a public competition where students were asked to participate. The goal was to prepare students to the competition.
- *App Challenge*. The main goal of App Challenge was to stimulate students in engaging in the project, as well as to have excitement throughout the course. The participation to the App Challenge was on voluntary base, namely students participated only if interested. In our case, all the students participated in the App Challenge. During the competition, students gave a demonstration (the imposed time limit was eight-minute) to a panel formed by external IT managers of national and

[3]http://www.zerottonove.it/unisa-grande-successo-per-la-prima-edizione-di-app-challenge/

TABLE I
THE PERCEPTION QUESTIONNAIRE

ID	Question
P1	Managing my project with GitHub was easy.
P2	Using the Software Configuration Management features offered by GitHub (e.g., commit, check-out) was easy.
P3	Using the communication features offered by GitHub (e.g., notification, tagging) was easy.
P4	I think that the app I developed is complex.
P5	Basing on your experience, the development of mobile app is easy with respect to traditional desktop applications.
P6	Basing on your experience, the development of mobile app is easy with respect to traditional web applications.
P7	My experience in the development of mobile app during this course was involving.
P8	The final competition was a stimulus for improving the quality of my work with respect to a traditional exam.
P9	After this course, my Android development competences are: ____

international companies, whose business included the development of apps for smart devices. Each team of students had to show that their app meets the market needs, explain which technologies they selected and why, discuss their choice on the User Interface, and present a live demonstration of the developed app. The first three projects received a prize from the jury composed of 10 IT managers. The first prize consisted in two iPads, the remaining were external hard disks. We asked the students to fill a grid scored from 1 to 10 concerning the following aspects: originality, business value, User Interface, technical quality, presentation. The prizes were assigned considering the results of the IT managers' evaluation. One of the main goal related to the organization of our App Challenge was to assess the students' apps from a professional perspective.

- *Students.* We were also interested in collecting some feedback from students about their perception in: using GitHub, developing apps for smart devices equipped with an Android operating system, and participating in App Challenge. To this end, we asked the students to fill in the questionnaire reported in Table I.
- *Software and project metrics*. We collected both software and project metrics [29]. A metric is a quantitative measure of a degree to which a software system and/or process possesses some property. We collected software and project metrics for two main reasons: *(i)* to assess team productivity and work and *(ii)* to study the value of these metrics to estimate the effort needed to develop mobile apps. For space reason, we will focus here only on the first point. We collected the following metrics:
  - *Requirements*, the number of functional requirements individuated in the RAD.
  - *Checkouts*, the number of local working copy taken from the repository by the team members. It provides an indicative idea of how much the team members are active and how their work is distributed;
  - *Time*, the time the students were active on the development phase of their project. It has been obtained by analyzing the activity log of GitHub;
  - *User Interfaces*, the number of graphical components composing the user interface of an app. In particular, we considered the number of XML file describing the Android activity user interfaces;
  - *LOC*, the number of lines of code, source code comment excluded.
  - *Cyclomatic Complexity*, a measure of the control complexity of a program. It measures the amount of decision logic in a source code function. It is a measure of how is structured a program. A high Cyclomatic Complexity denotes a bad structure and high risk of errors.
  - *Depth Inheritance Tree (DIT)*, which measures the software complexity of an inheritance hierarchy. It is the length of the longest path from a given class to the root class in the inheritance hierarchy. Some studies have shown that higher DIT rate corresponds with larger error density and lower quality [29]. The smaller the DIT, the more abstract and simpler the class would become, but decreases the class reusability. While the more a class inherits, the more difficult to understood the design is.

Our choice in selecting these metrics was mostly based on their simplicity in collecting and because they are well known and widely adopted (e.g., [29]).

Together with OO and traditional size code metrics, we also measured method calls in mobile apps. Method calls

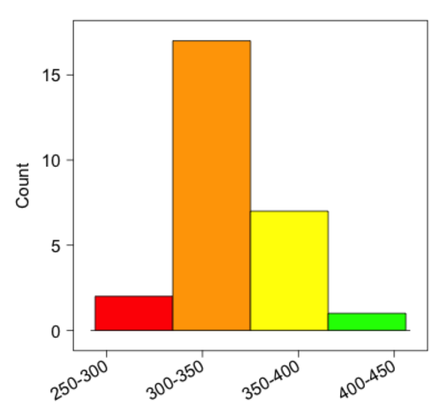

Fig. 2. The scores attributed by the jury.

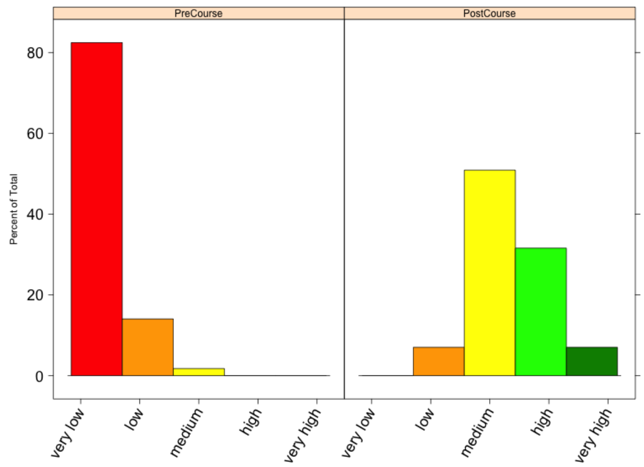

Fig. 3. The student perceptions on their Android development competencies Pre and Post course.

classified as both internal method and API (Application Programming Interface) calls. Internal method calls are invocations to methods the original developer implemented in the app, while API calls represent invocations to methods that Android provides. These metrics quantifies which use the app makes of native functionalities.

V. Results

All the teams completed their app and took part to the final competition. 11 apps were games, such as graphics retro-based games, or based on word guessing or math ability; 13 apps supported productivity (e.g. apps providing information on a City Hall, supporting people management or personal training), the remaining were social apps, e.g. for sharing their own travel diary or meeting people of interest.

A. App Challenge

The scores of the jury are graphically summarized in Figure 2. Descriptive statistics are also reported in Table II. The possible scores could range from 1 to 500. Thus, a mean score 331 with only two apps that scored less than 300 revealed a good opinion of the IT managers on the students' apps. The app considered the best obtained 405 as the score. The app was a very captivating game. The developers were very able to present and motivate their app, also performing a particularly suggestive spot. The User Interface was very simple and fascinating. The technical complexity was lower because the game mainly worked on the mobile device, except for the server-side score management.

B. Student perception

Concerning the opinions the students had on their Android development competencies, they perceived a notable improvement before and after the course, as shown in Figure 3. In particular, this figure depicts the histogram related the perceptions before and after the course, collected by the pre-course and the perception questionnaires (question

TABLE II
Jury evaluation statistics

Min	Max	Median	Mean	St. Dev.
282	405	328	331	27.17

P9), respectively. It is of practical interest to estimate the magnitude of performance difference perceived first and after the course. To this aim, we adopted the $Cohen\ d$ effect size. The effect size is considered negligible $d < 0.2$, small for $0.2 \leq d < 0.5$, medium for $0.5 \leq d < 0.8$, and large for $d \geq 0.8$). In our case (paired analyses), it is defined as the difference between the means (M_{POST} and M_{PRE}) divided by the standard deviation of the (paired) differences between samples σ_D.

$$d = \frac{M_{POST} - M_{PRE}}{\sigma_D}$$

Since the effect size is $d = 2.28$, we can consider that the students perceived that the course has had a considerable positive effect on their Android development competencies.

The answers to the perception questionnaire are graphically summarized in Figure 4. In particular, the greater part of the students asserted that GitHub eases the management of projects, 31 expressed a positive judgment (question P1). 36 students positively judged the CSM support offered by GitHub (P2), while 33 expressed a positive judgment on its communication feature (P3). 30 students judged complex the app they developed (P4). Most students (36) considered easier to develop mobile apps with respect to desktop ones (P5), while most of them considered easier develop web apps (P6). A high number of students (50) perceived the course involving (P7) and the final competition was very appreciate by 55 students (P8).

C. Project metrics

The values of the considered metrics are summarized in Figure 5. In general, the produced apps do not have a large number of functional requirements: the projects were characterized by median 8. The number of checkouts is not elevated (54 on average), probably because often students worked in pair programming modality, on the same PC. The time took to develop the app on average was 59 working days. The time to accomplish the analysis phase should also be added (about one month). The User Interfaces produced for each app were 33 on average. The number of Line Code (LOC) was on average 3609. Cyclomatic Complexity was on average 1.83, which denotes a good modular structure of the code (low risk of errors for values less than 10). DIT was 4 on average. This means that the classes are not much reused, i.e., teams develop for each functional requirements.

Table III reports the descriptive statistics for API (Application Programming Interface), internal method calls and the total number of calls. Internal method calls are invocations to methods the original developer implemented in the app, while API calls represent invocations to methods that Android provides. Half of the apps made more than 726 API

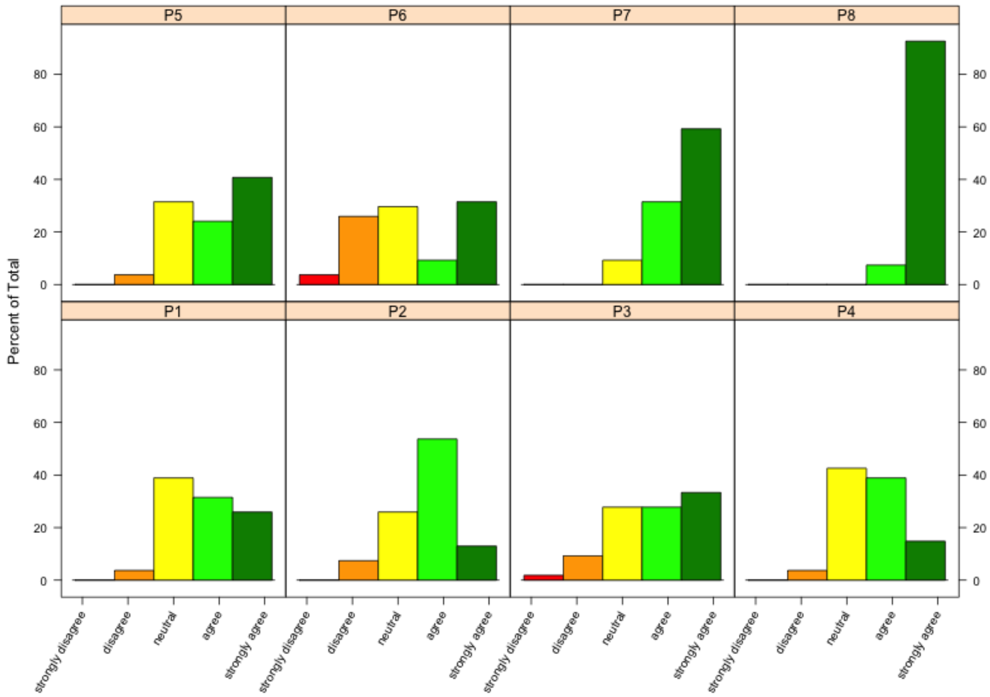

Fig. 4. The Perception Questionnaire results.

calls, that is they made a large use of the functionalities offered by the Android operating system.

VI. Conclusion and Discussion

In this paper, we have presented a teaching experience gained in the context of an academic course at the University of Salerno for the development of applications for smart devices. In such a course, students arranged in teams implemented apps and cooperated using GitHub. A competition to establish the best developed app was also conducted and the jury in charge of judging the apps was composed by IT professional managers.

The findings gained from our teaching experience can be considered positive: all the students delivered the projects on time, with a good level of quality and completeness with respect to the established requirements. The possible motivations could be related to the following aspects: first, all the students were enthusiastic in developing apps for smart devices; secondly, their activity was monitored thanks to the use of a GitHub which enabled continuous monitoring of the team work in all the phases of the development process, starting from the project proposal. Last but not least, let the students present their work to IT managers belonging to top IT companies. Indeed, by examining the project activity of the teams, when they knew of the company involvement their production notably increased. The lecturer and the tutors continuously motivated the students, also providing suggestions on the way they had to communicate. The App Challenge was successfully also because allowed the best students to be placed or to increase their familiarity with the work market. For example, a TLC company involved in this competition hosted the winning students for a stage because they demonstrated to be young talent with a strong ability to innovate. Many other students were required by the other companies involved in the App Challenge. Overall, all the companies manifested a positive judgment on the competition and on the work the students did. In fact, many of these companies asked to be informed and involved in future similar initiatives.

As future work, we plan to fully involve the IT professional managers in the next edition of the Mobile Application Development course. In particular, we would involve them as the role of coach. Future work will be also devoted to introduce in the next year course cloud platforms for implementing the back-end of the apps for smart devices.

TABLE III

Descriptive statistics on the API and internal method calls

Method call	Min	Max	Median	Mean	St. Dev.
API	36	11449	726	1398	2236
INTERNAL	9	1325	180	331	400

Acknowledgment

We thank all the students to this course and the IT managers, who judged the apps these students developed.

References

[1] Gartner, *Top 10 Mobile Technologies and Capabilities for 2015 and 2016*. https://www.gartner.com/doc/2665315, 2014.

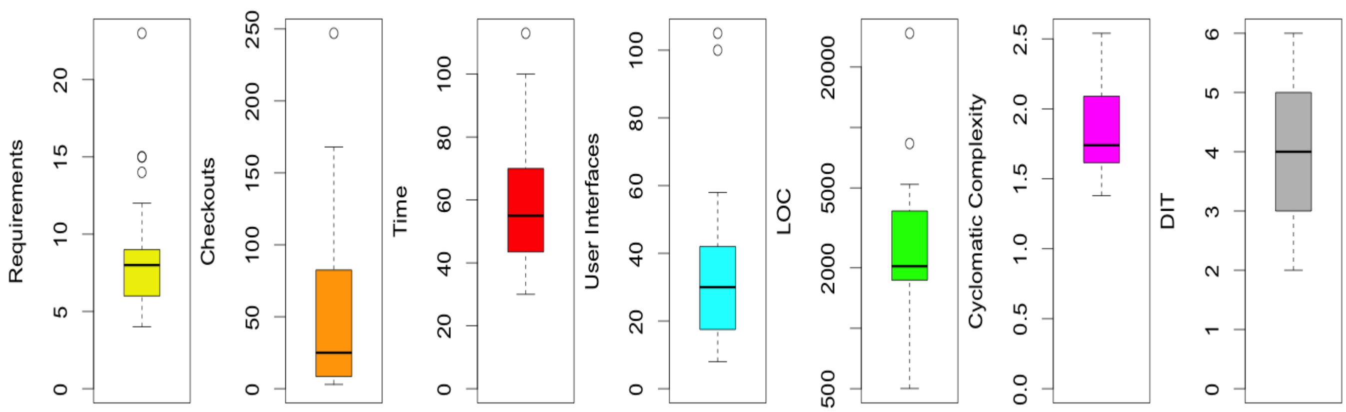

Fig. 5. The boxplots of the project metrics.

[2] R. Francese, M. Risi, G. Tortora, and G. Scanniello, "Supporting the Development of Multi-Platform Mobile Applications," in *Proceedings of the 15th IEEE International Symposium on Web Systems Evolution (WSE)*, pp. 87–90, 2013.

[3] R. Francese, M. Risi, G. Tortora, and M. Tucci, "Visual Mobile Computing for Mobile End-Users," *IEEE Transactions on Mobile Computing*, 2015.

[4] M. Cimitile, M. Risi, and G. Tortora, "Automatic Generation of Multi Platform Web Map Mobile Applications," in *Proceedings of the 17th International Conference on Distributed Multimedia Systems (DMS)*, pp. 84–89, 2011.

[5] R. Francese, M. Risi, and G. Tortora, "Management, Sharing and Reuse of Service-Based Mobile Applications," in *Proceedeings of the 2nd ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, pp. 105–108, 2015.

[6] A. Zagalsky, J. Feliciano, M.-A. Storey, Y. Zhao, and W. Wang, "The Emergence of GitHub As a Collaborative Platform for Education," in *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW)*, pp. 1906–1917, ACM, 2015.

[7] P. C. Blumenfeld, E. Soloway, R. W. Marx, J. S. Krajcik, M. Guzdial, and A. Palincsar, "Motivating Project-Based Learning: Sustaining the Doing, Supporting the Learning," *Educational psychologist*, vol. 26, no. 3-4, pp. 369–398, 1991.

[8] J. R. Savery and T. M. Duffy, "Problem Based Learning: An Instructional Model and its Constructivist Framework," *Educational technology*, vol. 35, no. 5, pp. 31–38, 1995.

[9] P. Dillenbourg, "What do You Mean by Collaborative Learning?," *Collaborative-learning: Cognitive and Computational Approaches.*, pp. 1–19, 1999.

[10] G. Scanniello and U. Erra, "Distributed Modeling of Use Case Diagrams with a Method Based on Think-Pair-Square: Results from Two Controlled Experiments," *Journal on Visual Languages and Computing*, vol. 25, no. 4, pp. 494–517, 2014.

[11] D. A. Norman and J. C. Spohrer, "Learner-Centered Education," *Communications of the ACM*, vol. 39, no. 4, pp. 24–27, 1996.

[12] J. Laffey, T. Tupper, D. Musser, and J. Wedman, "A Computer-mediated Support System for Project-Based Learning," *Educational Technology Research and Development*, vol. 46, no. 1, pp. 73–86, 1998.

[13] M. E. Joorabchi, A. Mesbah, and P. Kruchten, "Real Challenges in Mobile App Development," in *Proceedings ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pp. 15–24, ACM Press, 2013.

[14] S. Kizaki, Y. Tahara, and A. Ohsuga, "Software Development PBL Focusing on Communication Using Scrum," in *Proceedeings of the IIAI 3rd International Conference on Advanced Applied Informatics (IIAIAAI)*, pp. 662–669, 2014.

[15] S. Yadav and J. Xiahou, "Integrated Project Based Learning in Software Engineering Education," in *Proceedings of the International Conference on Educational and Network Technology (ICENT)*, pp. 34–36, 2010.

[16] Q. C. Lang, "Developing an Asynchronous Computer Mediated Communication Tool for Project-Based Learning," in *Proceedings of the 4th International Conference on Distance Learning and Education (ICDLE)*, pp. 222–225, 2010.

[17] J. Macias, "Enhancing Project-Based Learning in Software Engineering Lab Teaching Through an E-Portfolio Approach," *IEEE Transactions on Education*, vol. 55, no. 4, pp. 502–507, 2012.

[18] O. Ardaiz-Villanueva, X. Nicuesa-Chacón, O. Brene-Artazcoz, M. L. S. de Acedo Lizarraga, and M. T. S. de Acedo Baquedano, "Evaluation of Computer Tools for Idea Generation and Team Formation in Project-Based Learning," *Computers & Education*, vol. 56, no. 3, pp. 700–711, 2011.

[19] Y.-T. Lin, Y.-M. Huang, and S.-C. Cheng, "An Automatic Group Composition System for Composing Collaborative Learning Groups Using Enhanced Particle Swarm Optimization," *Computer & Education*, vol. 55, no. 4, pp. 1483–1493, 2010.

[20] C. A. Bowers, J. A. Pharmer, and E. Salas, "When Member Homogeneity is Needed in Work Teams: A Meta-Analysis," *Small Group Research*, vol. 31, no. 3, pp. 305–327, 2000.

[21] S. Hooper and M. J. Hannafin, "The Effects of Group Composition on Achievement, Interaction, and Learning Efficiency during Computer-Based Cooperative Instruction," *Educational Technology Research and Development*, vol. 39, no. 3, pp. 27–40, 1991.

[22] L. Williams and R. Kessler, *Pair Programming Illuminated*. Addison-Wesley Longman Publishing Co., Inc., 2002.

[23] C. McDowell, L. Werner, H. E. Bullock, and J. Fernald, "Pair Programming Improves Student Retention, Confidence, and Program Quality," *Communications of the ACM*, vol. 49, no. 8, pp. 90–95, 2006.

[24] GitHub. `https://github.com`.

[25] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository," in *Proceedings of the ACM International Conference on Computer Supported Cooperative Work (CSCW)*, pp. 1277–1286, ACM, 2012.

[26] P. Dourish and V. Bellotti, "Awareness and Coordination in Shared Workspaces," in *Proceedings of the ACM International Conference on Computer-supported Cooperative Work (CSCW)*, pp. 107–114, ACM, 1992.

[27] T. Gross, C. Stary, and A. Totter, "User-Centered Awareness in Computer-Supported Cooperative Work-Systems: Structured Embedding of Findings from Social Sciences," *International Journal of Human-Computer Interaction*, vol. 18, no. 3, pp. 323–360, 2005.

[28] A. Sarma, Z. Noroozi, and A. van der Hoek, "Palantir: Raising Awareness among Configuration Management Workspaces," in *Proceedings of the 25th International Conference on Software Engineering (ICSE)*, pp. 444–454, 2003.

[29] V. R. Basili, L. C. Briand, and W. L. Melo, "A Validation of Object-Oriented Design Metrics As Quality Indicators," *IEEE Transactions on Software Engineering*, vol. 22, no. 10, pp. 751–761, 1996.