Puzzle: A Visual-Based Environment for End User Development in Touch-Based Mobile Phones

Jose Danado and Fabio Paternò

CNR-ISTI, HIIS
Via Moruzzi 1, 56124 Pisa, Italy
{danado,fabio.paterno}@isti.cnr.it

Abstract. Despite the widespread usage of mobile devices there is a lack of environments able to allow end users to create applications directly in such devices. In this paper, we present the Puzzle framework, which supports a visual environment for opportunistically creating mobile applications in touch-based mobile phones. The user interface is designed to be usable for mobile users that do not use programming languages in their daily work as well as to motivate end users to playfully experiment and create applications. In particular, we report on its user interface, framework and evaluation.

Keywords: End user development, ubiquitous computing, mobile computing, authoring tools.

1 Introduction

The number of available applications for smartphones is rapidly growing, together with the number of users interested in top-range mobile devices. In 2011 alone 472 million smartphones were sold [1]. The complexity of mobile applications is also increasing because of the many possible preferences and contexts of use [2].

In this paper, we present a framework, named Puzzle, which considers these trends and enables end users without programming experience to create applications in mobile touch-based devices. There are various reasons for this type of proposal. Professional developers lack the domain knowledge that end users cannot easily convey when transmitting requirements for a new application, and regular development cycles are too slow to meet users' fast changing requirements [3]. End-user developers outnumber professional developers, thus it is important to develop End-User Development (EUD) tools, which are easy to learn and use, and to increase their quality and relevance for the users [4]. Furthermore, the Internet, and wide spread usage of mobile devices are potential tools to create a shift from the conventional few-to-many distribution model of software to a many-to-many distribution model. In summary, Puzzle aims to allow for a smooth adaptation in fast, mobile and dynamic environments where end users can easily adapt their mobile applications to newer requirements.

M. Winckler, P. Forbrig, and R. Bernhaupt (Eds.): HCSE 2012, LNCS 7623, pp. 199–216, 2012.

This paper considers two main aspects: the user interface (UI) of the environment and the framework. Regarding the UI, we aimed to design an easy to learn and use UI, which can be used in touch-based mobile devices, and copes with the limitations given by mobile use. Puzzle is based on the metaphor conveyed by jigsaw pieces to stimulate end users to combine functional building blocks. Building blocks exchange data between them to allow end users to explore possible combinations within their applications. The decision on adopting such metaphor was based on its usage on other EUD environments [5][6]. Puzzle adopts a higher level approach not just mimicking a traditional language through a graphical metaphor, but providing jigsaw pieces ready to be combined on the go, thus decreasing the learning curve and motivating users to explore and use it. Furthermore, jigsaw pieces were designed to facilitate users to combine them, and to solve errors and conflicts made during their combination. In order to allow Puzzle to be used across several mobile touch-based devices, it was implemented with established and widespread web technologies, namely HTML, CSS and Javascript.

Regarding the framework, we aim to create a framework that is flexible enough to integrate existing frameworks and technologies, and handle fast changes in the technologies used, both in the framework and the resulting applications. Mobile applications developed using Puzzle can exploit pre-existing applications or services, physical interactive devices and features of the smartphone. Starting from scratch or with available basic applications, users can define complex applications that better meet their needs.

Evaluation of the environment has included end users without programming experiences in two different stages of the Puzzle development. In particular, we evaluated the following contributions: *a)* The touch-based User Interface to support creation of applications in mobile touch-based devices, and *b)* The architecture to support creation, modification and execution in touch-based phones.

This paper is structured as follows: after discussing related work, Section 3 presents a sample application. Section 4 introduces the proposed approach for the development of mobile applications. Then, we present the environment UI, and the framework, followed by the description of the usability evaluation carried out. Lastly, we draw some conclusions and provide indications for future work.

2 Related Work

EUD environments to create context-sensitive applications for mobile devices mainly targeted desktop environments. In general, the main domains of desktop EUD environments targeting context-sensitive applications were tourism and virtual guides. Contributions range from support for a set of template applications for tourism [7] [8], domain-related content management to support guided tours [8] [7] [9], collaboration of different stakeholders [9], up to an EUD environment where the context is enriched through addition of calendar events; and EUD where the environment uses concepts such as: event-based rules, or workflow rules [10].

MIT App Inventor expresses the reasoning of building applications similar to Scratch [6], where a traditional programming style is performed by combining jigsaw pieces. In Puzzle, jigsaws are also used as a metaphor for the development of mobile applications. The contribution in Puzzle is that it hides programming constructs within the implementation of jigsaw pieces and focuses on the combination of interactive functionalities. Functionalities can then be combined through the jigsaw metaphor, which allows for reduced user's attention, lowered learning curve and motivates users to explore it. Instead of requiring the user to detail the UI and application logic, Puzzle allows users to focus on easy combination of functions that match their requirements. Furthermore, detailed customization of jigsaw pieces is possible for advanced users through access to implementation files.

Desktop EUD environments lack the advantages of enabling end users to create applications opportunistically in mobile scenarios. Recent advances in smart phones have enabled the creation of mobile EUD environments. Contributions for mobile EUD address: parameterization of the mobile terminal [11], frameworks to support mobile authoring and execution [12], mobile authoring tools [13] [12], creation of UIs through sketching or by adding interactive techniques in the touch screen [14].

Puzzle extends these contributions through combination of the jigsaw metaphor and a color help system inspired by the work of Cuccurullo et al. [13]. This proposal exploits the jigsaw metaphor to convey the notion of connecting jigsaw pieces, while the colors are used to provide intuitive cues to correctly connect the outputs of each jigsaw piece. Furthermore, the approach is general and can support different types of data for the inputs and outputs of jigsaw pieces. Puzzle also supports iteration among the interactive functionalities represented by the jigsaw pieces, access to web-services, native phone features and interactive physical objects. All the resulting applications are based on widely deployed web languages (e.g. Javascript, HTML5, CSS3) and protocols (e.g. HTTP) not requiring the addition of plugins to access native and external functions, and enabling users to customize or reuse existing powerful platforms [15].

3 A Puzzle Application

In this section, we provide an example application that can be developed with Puzzle. The application allows the user to interact with an Arduino board controlling a power outlet. Through the usage of Arduino, the application can interact with a power outlet controlling a lamp and measure its consumption for user information. First the application checks the status of the power outlet, afterwards, the user can switch the status, and finally the switch is applied to the power outlet. The Puzzle framework allows end users to develop applications able to easily control physical objects with a touch-based phone, and provides an easy framework to create or modify the application when required. In our example, the Puzzle framework can also be used to control the consumption of the power outlet through another application by reusing Puzzle jigsaw pieces.

Jigsaw pieces are used to convey interactive functions available in Puzzle. Our example requires an end user to drag three jigsaw pieces to the center of the screen and connect them, namely “check lights”, “switch”, and “change lights” jigsaw pieces (see Fig. 1). The first jigsaw checks if the lights are on or off, the second jigsaw switches the value, and the last jigsaw sends the value to the lights. Later, the end user could modify the application by erasing the last two jigsaw pieces and connect “check lights” with a consumption display jigsaw piece to visualize the consumption in the power outlet.

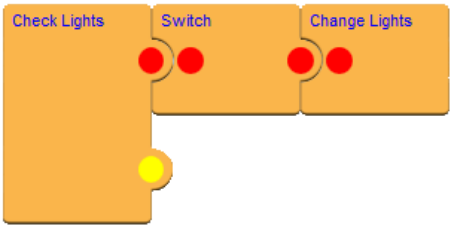

Fig. 1. Sample Puzzle Application

End users need to handle and connect jigsaw pieces as well as their connectors and types. Jigsaw pieces represent building blocks, which are able to receive outputs from other building blocks and, similarly, they output their results into other building blocks. Each building block is responsible for handling inputs and adapting them, if required, to enable adequate execution.

4 The Proposed Approach

Puzzle is a framework to support creation and execution of applications in a touch-based mobile phone. The framework is intended for users without knowledge on programming languages to playfully experiment supported functions. Puzzle provides: *a)* A jigsaw piece metaphor to convey a top-down left-to-right flow of data; *b)* Drag-and-drop interaction techniques for creation and modification of applications; *c)* A color help system to convey possible connections between jigsaw pieces; *d)* A help system in order to overcome usage doubts; *e)* Sliding and popup menus to save screen space.

The design of touch interaction in Puzzle is based on object selection, dragging and sliding. The user is able to drag jigsaw pieces into the working area, select jigsaw pieces or slide a previously hidden object. The limited screen space requires techniques to show relevant tasks and hide unused objects on the screen. For such purpose, sliding mechanisms, such as scroll bars and sliding menus are often used in its UI.

The framework generates web-based applications, and it is a web-based application as well. The motivation for this decision is based on the fact that updates can be easily performed in the framework and propagated to all clients. Furthermore, features such

as CSS3 media queries, or server-side techniques can deliver CSS files based on the requested browser features in order to better adapt Puzzle to the current mobile device.

5 User Interface

The Puzzle UI is divided into 3 modules, namely main, authoring tool and execution environment, as seen respectively in Figure 2. This UI is the result of the evaluation of two previous versions, which we report in the evaluation section.

Fig. 2. a) Main b) Authoring Tool c) Execution Environment

Firstly, users are introduced to a set of applications already available in the framework (Fig. 2.a). In the Main Module, users can create a new application from scratch, or execute one of the shown applications. An icon is presented for creating an application and the remaining icons allow the execution of a related application. The icon with a plus sign and a jigsaw piece in the top left corner is used to identify the creation of an application. Next, the selection of a new application directs the user to the Authoring Tool.

In Authoring Tool, users are able to create or modify an existing application (Fig. 2.b). A Puzzle application is developed by drag-and-drop jigsaw pieces to the center, and connecting them top-down left-to-right in order to obtain the composed functionalities. One single jigsaw piece can be executed without being connected, so as to allow the end user to explore its function.

Finally, the Execution Environment enables the execution of a selected application (Fig. 2.c). In the Execution Environment, the UI presented is defined in the implementation of the jigsaw pieces in execution. Additionally, a left menu is overlaid so that users are able to go back to the Main module or edit the current application in the Authoring Tool.

5.1 Authoring Process

The authoring process in Puzzle evolves in three steps: Authoring, Properties, and Execute. Figure 3 is a workflow chart showing their temporal evolution from left to

right, where rounded rectangles represent process phases, rectangles represent the actions allowed in each phase, and dotted rectangles represent stored data.

Creating an application directs the user to the authoring phase. In this phase, the user is able to add and connect the required features to the application. The available operations are: view a list of categories of jigsaw pieces, drag and connect jigsaw pieces, configure a jigsaw piece or delete it. Configure a jigsaw piece means customizing values of a jigsaw piece for a particular purpose. All the data related to jigsaw pieces are retrieved from the Building Block Repository. Afterwards, if the user is creating a new application, she is able to set its name and/or icon, and immediately pass to the last phase where the application is deployed and executed. The deployment of an application is performed through the application repository where all applications are stored.

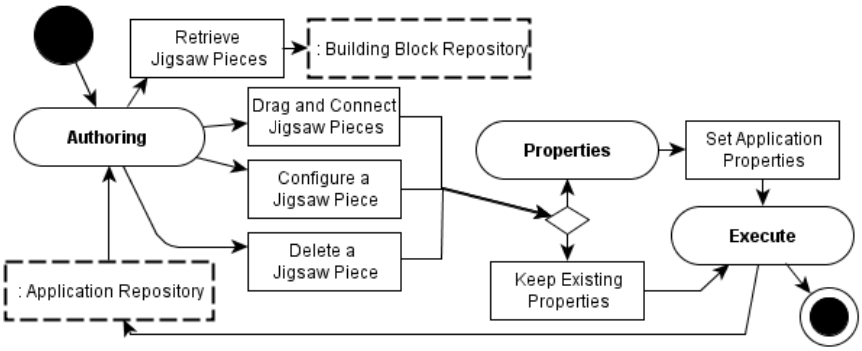

Fig. 3. Authoring Process Flowchart

5.2 Authoring Tool

The Authoring Tool has been designed taking into account the limited screen size and the interaction techniques available in mobile devices. The creation of a Puzzle application focuses on the usage of the 'jigsaw' metaphor. A jigsaw piece evokes the intuitive suggestion of assembling pieces together. Essentially, we allow users to connect building blocks and compose various arrangements through a series of top-down left-to-right couplings of pieces. Constraining connections in a left-to-right fashion provides users with the sense of a pipeline of information flow.

Each jigsaw piece has connections, named inputs or outputs. Inputs are placed in the left side of the 'jigsaw piece' and evoke the ability to receive a connection from another jigsaw piece. At the view level, inputs have an inner circular shape suggesting that they will receive a connection. At the implementation level, inputs are values received from connecting jigsaw pieces and such values are used within execution of the current jigsaw piece. Outputs are placed at the right side of the jigsaw piece and evoke the ability to connect into other jigsaw pieces. At the view level, outputs have an outer circular shape suggesting that they will send information. At the implementation level, outputs are values resulting from execution of the current jigsaw piece that can be used in connected jigsaw pieces.

In Puzzle, inputs and outputs are augmented with a color system used to indicate the possibility to connect two different jigsaw pieces only if the colors from the input and output of the jigsaw pieces match. We assumed that colors could be easier to distinguish when compared with a number of shapes to uniquely describe all matches. The function provided by a jigsaw piece is communicated to the user through a label (or icon) at the top center of a jigsaw piece (see Figure 4).

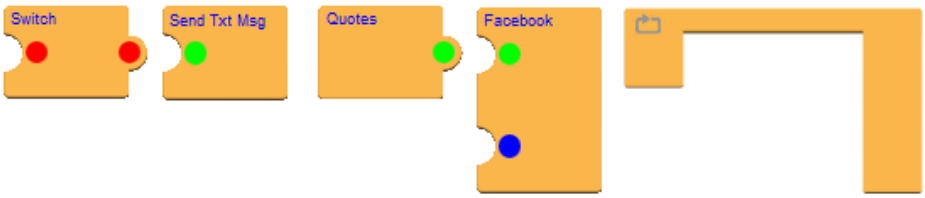

Fig. 4. Sample jigsaw pieces in Puzzle

The ability to select a jigsaw piece to use in a new application is provided to the user through a carousel, at the bottom, listing available jigsaw pieces in a category (see Figure 2.b). Puzzle has several categories of jigsaw pieces and the framework is not limited in the number of jigsaw categories as categories and jigsaw pieces can be added to the Building Block Repository (see Figure 5). Access to further categories is available through a tab icon on the left that slides a menu with the available list of categories. Currently, Puzzle has 5 categories and 10 jigsaw pieces. For example, included jigsaw pieces enable the application to: include a gallery of images from Flickr, post a message on Facebook, send a text message to a contact in the phone, list text messages, or switch lights connected to an Arduino-based board. Available jigsaw pieces were created to demonstrate the potentialities of the Puzzle framework not limiting the available functions as they can be extended.

The tabbed menu enables the user to access: the list of categories, common functions of the Authoring Tool, and execute an application (see Figure 2.b). The motivation for the usage of a tabbed menu is to allow common functions to be easily accessible while further details can be accessed and used on a sliding menu including available options related to the clicked tabbed icon. Additionally, screen space is saved for editing the application. The execute function was made available on this menu for its easy access.

The Authoring Tool also includes a help system. A help button is shown at the top of the screen if the user does not perform any interaction with the Authoring Tool for a certain amount of time. The motivation for this solution is to enable contextual tips when a user is blocked and also not disturb the user in case she has moved her attention to some other task (see Figure 2.b).

Further help on jigsaw pieces is also provided to end users. At the bottom carousel, a user can tap a jigsaw piece in order to trigger a popup menu with further information. If a jigsaw piece is at the center of the screen, a single tap on the jigsaw piece allows the user to raise a popup menu to further configure, edit or get help. The help provided is a description of the available functions of the jigsaw piece. At the

bottom right corner of the screen, end users can find a trash can, which allows them to remove a jigsaw piece from an application.

In the Authoring Tool, end users are also able to view the data flow. The data flow is managed by the framework and used within jigsaw pieces tasks. The available jigsaw pieces implement tasks, such as those in the example application to change the state of a power outlet. The functions should be easily identified by a non-programming end user and easy to learn and use. A motivation for such approach is that mobile users may have a reduced attention while using Puzzle, or may not be familiar with programming control flow operations. The support of iteration among functionalities represented by various jigsaws has been added in Puzzle since it was requested by some users. For example, while controlling a light or power outlet, users may require the application to loop until they exit the application.

5.3 **Execution Environment**

The execution environment executes one jigsaw piece each time, and stores and retrieves input/output values as required by each jigsaw piece. As a result, the UI for an application is the sum of the rendered jigsaw pieces UIs. Currently, each jigsaw piece has its own UI, created at development time.

The execution environment is similar to a mobile browser, with the additional possibility for the applications to access native phone features without requiring the installation of plugins. During execution, users are also presented with one tab icon that once clicked allows them to go to the Authoring Tool to edit the current Puzzle application or to the Main module (see Figure 2.c).

6 Framework

Contributions of the framework are the ability to: provide access to native and remote functions using web-based technologies, store and provide complex operations ready to be combined and used in an application, and allow non-programing end users to explore and use different technologies from their mobile devices.

The motivations to use web technologies for the implementation of the Puzzle framework were the elimination of administrative tasks such as software installation and update, and the support of a network of connected users. In addition, Puzzle includes a native module to access native mobile functions, which can be accessed through HTTP requests; and a server module to allow access for interactive physical objects connected to the framework (see figure 5).

Native phone functions allow the user to integrate information and features that are usually not included in web-based applications within her application. Interaction with physical objects allows end user developers to create applications that interact with embedded sensors and actuators and also foster user exploration of the possibilities technology can provide. Puzzle allows users to use, combine and remix data from multiple sources while gathering contributions from all users. Thus, we can expect Puzzle to benefit from an “architecture of participation” to which users can contribute with applications and, possibly, building blocks.

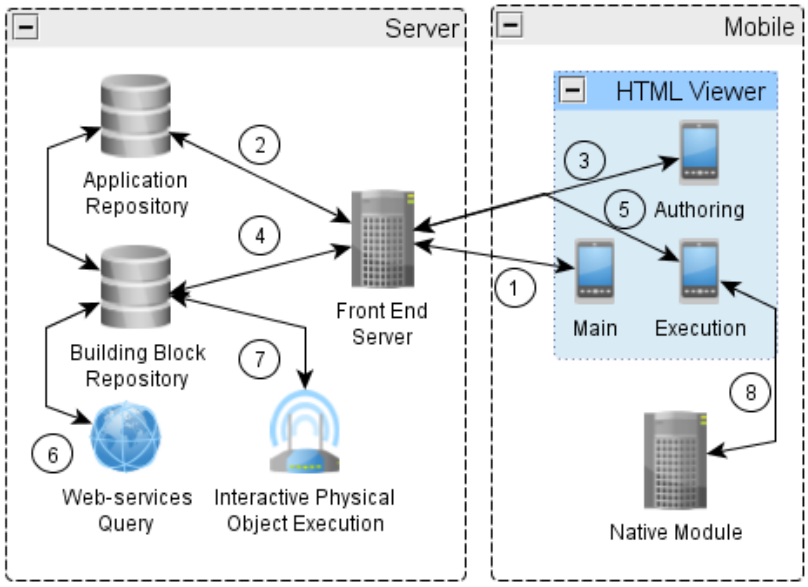

Fig. 5. Puzzle Architecture

Figure 5 describes the Puzzle architecture. The authoring tool, applications and building blocks are stored and assisted by the server managing the framework. The mobile side of the architecture contains a native application (Android) including an HTML viewer and a native module accessible through HTTP requests.

The flow of information starts in the Main module from the HTML Viewer. Once Puzzle starts, the Main module requests the list of available applications (connection 1) from the Front End Server. Such list is stored within the Application Repository and delivered to the Front End Server through an SQL request (connection 2). When an application is created from scratch, end users are directed by the Front End Server to the Authoring module (connection 3). Next, the Authoring module further requests to the Front End Server for available building blocks. Such information is stored in the Building Block Repository (connection 4). Afterwards, the Authoring module can request configuration details and further information from the Building Block Repository. When the application is created the Authoring module sends the information about the application to the Front End Server (connection 3) and stores that information in the Application Repository (connection 2).

Next, the end user is directed to the Execution module by the Front End Server while executing the previously created application (connection 5). During application execution, a building block can further request some external services, for example Flickr or Facebook (connection 6). Additionally, a building block can require interaction with physical objects, through connection 7, or request the Native module to execute native functions (connection 8).

7 Usability Evaluation

The goal of the framework is to allow end users without an IT background to create mobile applications opportunistically in touch-based devices. In order to address that goal, we targeted three sub-goals: *a)* the UI has to address the limitations of its usage on touch-based mobile devices, namely the screen size; *b)* the UI has to reduce the cognitive effort of end users without an IT background through adequate metaphors and interaction techniques; *c)* the framework has to support seamless tasks accomplishment. Thus, the framework should not require the end user to learn a programing language to create an application.

In an iterative process, creation and development of Puzzle used early prototyping to evaluate and develop design solutions and to gradually build a shared understanding of the needs of the end users as well as their possible future practices [16]. In this paper, we report on the evaluation of two previous prototypes targeting end users without an IT background, through the real usage of such prototypes (see Figure 6). In both evaluations, no tutorials or learning steps were provided to end users in order to evaluate if the UI was able to guide them through the creation of a mobile application. Furthermore, all sub-goals were addressed in both evaluations. Contributions from previous evaluations were also included in subsequent versions and newer issues of the UI were also tested. The mobile phones used within the evaluations were Android-based devices, namely using Android 2.3.

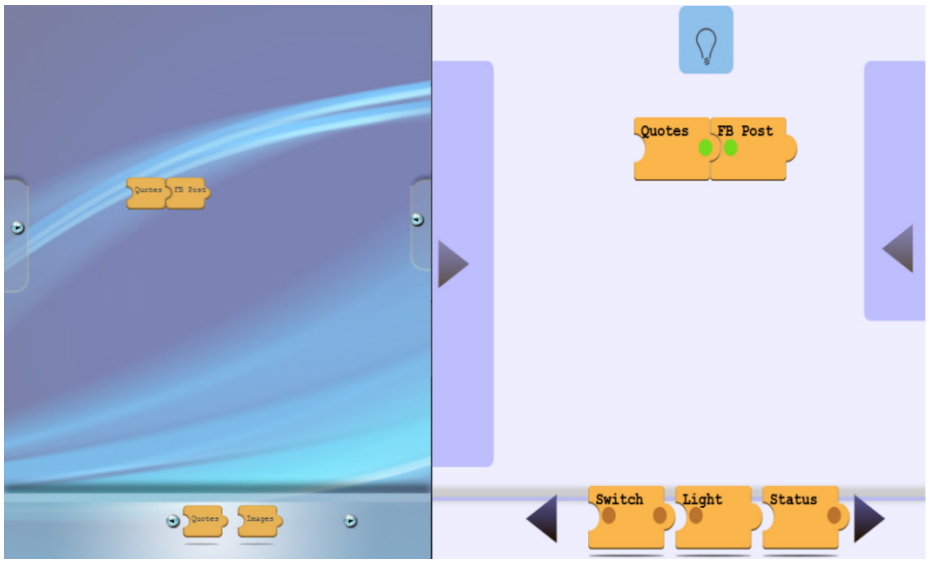

Fig. 6. a) First Prototype b) Second Prototype

The first user evaluation was performed with six mobile Internet users without an IT-related job. This evaluation focused in the ability to create mobile applications in a mobile device through the jigsaw piece metaphor, sub-goal *b)*. The group of users was also gender balanced with equal representation of both genders. Puzzle was evaluated in a relaxed home environment and the volunteers ranged from 26 up to 35 years old with an average of 31.16 years old and their jobs related to law practice, municipality administration or other municipality activities. The evaluation was conducted by one researcher involving two data collection methods: observation and questionnaire. Each volunteer was individually interviewed, where each session ranged from 40 up to 50 minutes, and the goals of the evaluation were clearly stated to volunteers.

For the second user test, a similar approach was considered. This evaluation focused on: the limitations of the UI regarding touch-based mobile devices, sub-goal *a)*; reducing the cognitive efforts of the user, sub-goal *b)*; and demonstrate that the framework could support the performance of envisioned tasks, sub-goal *c)*. Evaluations took place in an office with 7 users ranged from 31 years old up to 59 years old with an average of 44.1 years old. 5 users were females and 2 users were males. Furthermore, their jobs were not IT related, mainly in the administration of a research institute. As in the first evaluation, two data collection methods were used: observation and questionnaire. In addition, the test duration was similar and, after a general view over the framework, no further instructions were provided to test users.

7.1 **Method**

In both tests, the user evaluation was performed with a working prototype, so that users could realistically provide feedback on the usage of the framework. In detail, user evaluations included an introductory questionnaire, a set of tasks to accomplish, and a post-test questionnaire.

The introductory questionnaire included: *a)* Demography information, *b)* Previous end user development experiences, *c)* First evaluation of Puzzle authoring tool, and *d)* Which target users would be envisioned for such tool.

This step was performed to categorize the test users and collect data on their previous experiences. After the introductory questionnaire, users were asked to complete a set of tasks using Puzzle. The tasks included: *a)* Testing the usability of the left sliding menu to return to the list of applications (during execution), *b)* Testing the usability of the left sliding menu for editing an application and the drag and drop technique to compose an application, *c)* Configuration or deletion of a jigsaw piece while creating or reusing an application, *d)* Creation of a new application to show a map with the user's location and post it on Facebook, execute the application, and return to the list of applications. These tasks focused on sub-goal *b)*.

The test involved the basic functions available in Puzzle, namely sliding menus, drag and drop interaction techniques and the jigsaw piece metaphor for application creation and modification. Finally, a post-test questionnaire was performed. In the first user test, the questionnaire included a qualitative analysis about: *a)* Overall user evaluation of Puzzle, *b)* Evaluation on the fun, pleasure and ease of use, *c)* User's ability to create applications through that process, *d)* Best and worse characteristics on Puzzle, *e)* User's ability to use Puzzle, *f)* Type of applications that the user would

enjoy to create, *g)* Interest on controlling home appliances through usage of the tool, *h)* General comments and remarks.

In the second user test, the introductory questionnaire was similar to the one on the first user test. However, the tasks to be performed included: *a)* Creation of a simple application and evaluation of the color-based connection system and help system; *b)* Creation and modification of an application combining jigsaw pieces where a message gallery was used to select a message to be sent over a phone text message or into Facebook; *c)* Creation of an application to control lights from the mobile touch-based device; *d)* Suggestions to support iteration and selection in the framework.

The information collected in the first user test regarding sub-goal *b)* was used to improve the second prototype and newer functions were included to test sub-goals *a)* and *c)*. In detail, the second user test intended to evaluate a color-based connection system and a help system to reduce the cognitive effort. The second user test also evaluated the ability to use Puzzle to create applications integrating web services, native mobile functions, and interactive physical objects. The post-test questionnaire included a quantitative and a qualitative analysis. The quantitative analysis included 18 sentences and the user was asked to evaluate each sentence within a Likert scale from 1 (Strongly Disagree) to 5 (Strongly Agree). The purpose of this set of sentences was to support the analysis of the following aspects: learnability, efficiency, effectiveness, memorability, errors and satisfaction. The qualitative analysis of the post-test questionnaire was similar to the one in the first user test.

7.2 **Results**

The results showed that the UI could be used with a low cognitive effort and addresses the limitations of touch-based mobile devices. In addition, the framework is also able to include, integrate, and execute the relevant tasks. Namely, the jigsaw piece metaphor, as presented, was easy to use and combine; and the framework eased integration of web-services, native phone functions and interactive physical objects. First, we present the results of the first user test followed by those of the second test. The results are presented first for the introductory questionnaire, next for the set of tasks to accomplish, and finally for the post-test questionnaire.

In the first introductory questionnaire, two users had no previous experiences regarding end user development. One user had developed simple web pages through Microsoft FrontPage. Four users had modified databases in Microsoft Access. Three users had used formulas or macros in Microsoft Excel. Finally, one user reported customization of MS-DOS batch files.

Remaining results on the first introductory questionnaire can be seen in Table 1. The first column describes the question, the second column indicates the number of users who gave the corresponding positive comments regarding the question, the third column indicates the number of users who gave the corresponding negative comments regarding the question.

After an introductory questionnaire, test users were required to accomplish a set of tasks. In the first user test, users were asked to perform four tasks. Success on completion for each task was divided in 3 categories: 0, not completed; 1, completed with difficulty or help; 2, easily completed. Observed comments were also registered.

Table 1. First introductory questionnaire results

	Positive		Negative	
Questions	**#**	**Comments**	**#**	**Comments**
First Evaluation	6	Combinations of jigsaw pieces	2	Doubts on how to combine jigsaw pieces
Target users	2	Public use	4	Professionals
Satisfaction	6	Simple, pleasurable, and easy to use	0	

In the first task, the left sliding menu in execution was successfully used by five users and one required some help. Expressed concerns targeted the size of the menu. In the second task, users were easily able to drag jigsaw pieces to the center of the screen to create a mobile application. Five were able to create an application easily (2) while one required minor help (1). Suggested improvements relate to usage of different icons for creation and execution, enlarge icons and symbols, improve abbreviations, use double tap to execute an application and add help to Puzzle. In the third task, all users were able to easily identify the popup menu to configure or delete a jigsaw piece (2). The last task included combining jigsaw pieces within an application. In this task, four users were able to perform the task (2) while two required some minor help (1). Concerns in the task were related to an unclear flow of data between jigsaw pieces and the abbreviations in jigsaw pieces. It was also suggested the use of images instead of text.

Finally, a post-test questionnaire was presented to users. In the first user test only a qualitative analysis was performed. Table 2 shows the results for the first post-test questionnaire. The first column describes the question, the second column describes the number of users with positive comments regarding the question, the third column describes the number of users with negative comments regarding the question. Positive and negative expressions were added for each user. In cases where the sum of answers is more than 6, it means that there were users with both positive and negative comments regarding the question.

Table 2. Tasks performed in the first user test

	Positive		Negative	
Questions	**#**	**Comments**	**#**	**Comments**
Overall view	6	Easy and simple, useful	1	Can be improved
Satisfaction	6	Pleasurable and fun to use	2	Icon and button sizes, Add double tap to execute, improve dataflow
Creation Process	5	Intuitive and easy	1	Allow for customization, add jigsaw pieces
Best and worst	6	Useful, ability to control sensors and actuators	6	Customization, improve intuitiveness, proactive help, execution errors
Usage	5		1	Prefer existing apps
Envisioned applications	5	Smart home, social networks, work support		
Comments	2	Domain experts and professionals		

In the second user test and in the introductory questionnaire, two users had no experience with end user development, four had some experiences with Microsoft Excel focusing on formula's usage, and one was using tools for web development such as Joomla, or Wordpress. Similar to the presentation of results for the first user test, Table 3 shows the results for the introductory questionnaire.

Table 3. Second introductory questionnaire results

	Positive		Negative	
Questions	**#**	**Comments**	**#**	**Comments**
Willingness to create applications	3	Fun, leisure or improve tasks	4	Lack of motivation
Initial evaluation	7	Pleasurable and fun to use	0	
Target users	3	Public use	4	Professionals / self learners
Satisfaction	7	Simple, easy and pleasurable	0	

In the second user test, the set of tasks was changed according to the outcomes and improvements gathered from the first user test. Furthermore, completion of the tasks was also updated to a scale of 1 (Not completed) up to 5 (Easily completed). The first task included testing the color and help system during the development of an application. One user performed the action easily (5), four required minor help (4), and one required further help with language issues and finding execution (3).

The second task was the development of an application to send a text message from a message gallery and modify it to post on Facebook. One user performed the task easily (5), two required help on language issues (4), and four users required help on describing the flow of information (3). In this task, users suggested adding a trash to ease the deletion of a jigsaw piece. The third task involved development of an application to control a set of lights with Arduino. Two users completed the application easily (5), four required minor help to clarify the actions to be performed (4) and one user was not able to understand some functions and the observer helped to accomplish the task (2).

The last task included suggestions for selection of jigsaw pieces and support for iteration. Selection was preferred by making a circle around jigsaw pieces. Other approaches included taping all jigsaw pieces or perform a diagonal selection. Iteration was preferred through a special jigsaw piece connecting the initial and the final jigsaw piece. Other options included selecting jigsaw pieces for iteration and configure iteration on a popup menu.

Next, a quantitative and qualitative analysis was performed through questionnaires. The quantitative analysis evaluated learnability (questions 1-6), efficiency (7-9), effectiveness (10-12), memorability (13), errors (14-16) and satisfaction (17-18) while using Puzzle. Box plots from the user evaluation using a scale of 1 to 5 Likert scale targeting these 18 sentences are listed in figure 7. The Box plot includes the smallest observation (sample minimum), lower quartile (Q1), median (Q2), upper quartile (Q3), and largest observation (sample maximum) for each question. Bottom and upper tickers represent minimum and maximum, respectively. Bottom and upper lines for

the orange boxes represent Q1 and Q2, respectively. The upper line of the blue box is the Q1. The space between limits of the orange and blue boxes helps to indicate the degree of dispersion (spread) and skewness in the data, and identify outliers. The qualitative analysis is listed and was evaluated similarly to the first user test and results are listed in table 4.

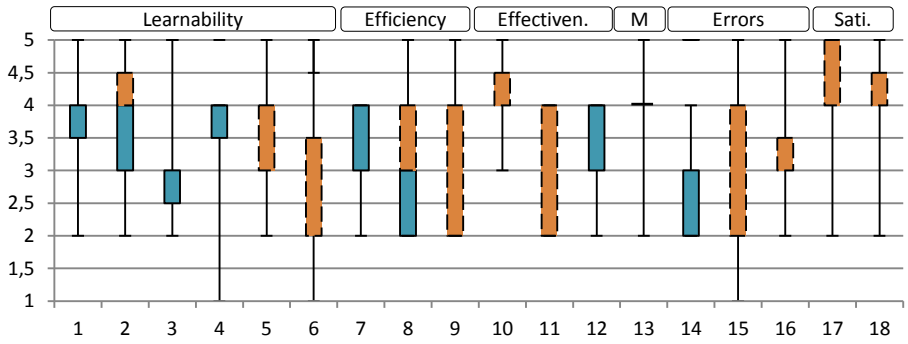

Fig. 7. Quantitative analysis results

Table 4. Tasks performed in the second user test

Questions	Positive #	Positive Comments	Negative #	Negative Comments
Overall view	7	Good and pleasant	0	
Satisfaction	7	Good and simple	2	Icon and button sizes, Add double tap to execute, improve dataflow
Creation Process	5	Intuitive and easy	2	Allow for customization, add jigsaw pieces
Best and worst	7	Simple UI, ability to integrate web-services, native phone features and interactive physical objects	5	Customization, improve intuitiveness, proactive help, execution errors
Usage	3	Public Use	4	Professionals and/or self learners
Envisioned applications	5	Home automation, social network integration, phone features, support for common everyday tasks		

7.3 Discussion

Puzzle evaluation validated a touch-based UI and a framework supporting end users without an IT background to create applications in the mobile device. The jigsaw piece metaphor was easily understood and applied to develop an application. The first evaluation focused on evaluating the metaphor and the results confirmed that end users were able to use it for manipulating high level functions. The UI also copes with the limitations of touch-based mobile devices, where both evaluations addressed such limitations. The size of UI components, their representation, and interaction techniques used have been evaluated to better address end users expectations. The drag-and-drop interaction technique and the bottom carousel are keys to the Puzzle UI

and proved to be effective. The evaluation also demonstrated that the framework can support creation, modification and execution of applications in the mobile device.

In particular, the results regarding the UI were positive in terms of easiness to learn, efficiency, effectiveness, easiness to remember, and number of errors . Figure 8 shows the means (from the data in Fig. 7) related to the questions associated with the corresponding aspects within a Likert scale from 1 to 5.

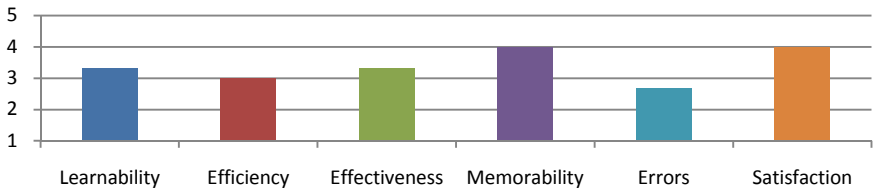

Fig. 8. Quantitative analysis average

In the first user test, a central goal in the evaluation was the assessment of using a jigsaw piece for development of applications. Even if this approach does not provide a WYSIWYG view, users were easily and immediately able to test their applications on the platform. Furthermore, the drag-and-drop interaction technique proved to be effective to add jigsaw pieces into the center of the screen and create an application. The next step was to combine jigsaw pieces and specify the data flow in an application. For that purpose, four users easily understood how to combine jigsaw pieces and how the data was flowing. However, two users pointed that the framework could easily convey a correct order to combine jigsaw pieces and correctly build the desired application.

Concerning the type of applications that users would like to develop within Puzzle, social applications and control of sensors and actuators were the main applications selected. The smart home scenario was the most mentioned, possibly integrating social networks. Users were also interested in applications that could react to previously identified contextual scenarios. As an example, enable the possibility to develop applications that turn the heating system or a hoven when the user is going home.

For the second user test, we adapted the Puzzle UI to accommodate an increase in icons and buttons size. In addition, we have restructured common UI controls for consistency between editing and execution, and simplified the UI. We also added a help system to Puzzle, so that provided hints could help users to proceed. Such hints disappear after being checked to save screen space. Finally, we added a color system to ease connection of jigsaw pieces. Concerning newer functionalities, we included newer jigsaw pieces to envision applications that took advantage of web-services, native phone features and interactive physical objects.

For such refinements, users further suggested to adjust the text size, icons and symbols, inclusion of a trash can in order to make the deletion task more intuitive, better convey the data flow in connected jigsaw pieces, and improve the color system

to better indicate its purpose. In the user tests, suggestions for selection highlighted circling with the finger around objects as the preferred interaction technique. Iteration was proposed by adding a different jigsaw that would connect the begin and the end of a set of jigsaw pieces involved in the iteration.

The quantitative analysis (see Figure 6) indicates that Puzzle is easy to use and learn with improvements required for the color and help system. The metaphors used are also easily understood in terms of their meaning and usage. However, results were affected by the jigsaw snap function to connect pieces. In occasions, snapping was not working smoothly causing some problems. Puzzle is also effective as the result of functions and applications are what users expect. Indeed, functions in Puzzle are very easy to memorize as editing of an application was not causing any problems to users. The detected errors were mainly due to connecting jigsaw pieces and the snap algorithm. Improvement on the labels describing jigsaw functions is also considered.

The qualitative evaluation reinforced the positive feedback on using Puzzle and the adequacy of the metaphors and interaction techniques used. Users liked the ability to integrate different features seamlessly and further suggested improvements, namely the ability to pan, zoom and have a general view over a more complex application. The users highlighted that it would be interesting to navigate within the jigsaw pieces while developing an application with a similar interaction technique as with current map navigation.

8 Conclusions and Future Work

Puzzle targets users without programming skills willing to start developing mobile applications on their touch-based phones and further execute such applications in the touch-based mobile phone. From our results, Puzzle is addressing the initial goals and users can easily create mobile applications from scratch without a previous learning phase. Future work will include testing iteration constructs on Puzzle applications, based on the user suggestions; and adding a selection mechanism so that jigsaw pieces could be grouped and new jigsaw pieces created based on that grouped function. This would allow users to group common functions and use them in the future. Furthermore, a zoom, pan and overall view can be added to allow navigation within an application as well as allow the user to see the details of grouped jigsaw pieces. In addition, future improvements on Puzzle will include improvements in the color and help system to further support users to combine jigsaw pieces.

References

1. Gartner: Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth. Press Release, Gartner, Egham, UK (2011), http://www.gartner.com/it/page.jsp?id=1924314
2. Gronli, T.-M., Hansen, J., Ghinea, G.: Android vs Windows Mobile vs Java ME: a comparative study of mobile development environments. In: Makedon, F., Maglogiannis, I., Kapidakis, S. (eds.) Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments (PETRA 2010), Article 45, 8 pages. ACM, New York (2010)

3. Lieberman, H., Paternò, F., Klann, M., Wulf, V.: End-User Development: An Emerging Paradigm. In: Lieberman, H., Paternò, F., Wulf, V. (eds.) End User Development, vol. 9, ch. 1, pp. 1–8. Springer, Dordrecht (2006)
4. Boehm, B., Abts, C., Brown, A., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D., Steece, B.: Software Cost Estimation with COCOMO II. Prentice, Upper Saddle River (2000)
5. App Inventor MIT (2012), http://info.appinventor.mit.edu/
6. Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., Kafai, Y.: Scratch: programming for all. Commun. ACM 52, 60–67 (2009)
7. Ghiani, G., Paternò, F., Spano, L.D.: Cicero Designer: An Environment for End-User Development of Multi-Device Museum Guides. In: Pipek, V., Rosson, M.B., de Ruyter, B., Wulf, V. (eds.) IS-EUD 2009. LNCS, vol. 5435, pp. 265–274. Springer, Heidelberg (2009)
8. Hull, R., Clayton, B., Melamed, T.: Rapid Authoring of Mediascapes. In: Davies, N., Mynatt, E.D., Siio, I. (eds.) UbiComp 2004. LNCS, vol. 3205, pp. 125–142. Springer, Heidelberg (2004)
9. Celentano, A., Maurizio, M.: An End-User Oriented Building Pattern for Interactive Art Guides. In: Costabile, M., Dittrich, Y., Fischer, G., Piccinno, A. (eds.) IS-EUD 2011. LNCS, vol. 6654, pp. 187–202. Springer, Heidelberg (2011)
10. Realinho, V., Eduardo Dias, A., Romão, T.: Testing the Usability of a Platform for Rapid Development of Mobile Context-Aware Applications. In: Campos, P., Graham, N., Jorge, J., Nunes, N., Palanque, P., Winckler, M. (eds.) INTERACT 2011, Part III. LNCS, vol. 6948, pp. 521–536. Springer, Heidelberg (2011)
11. Tuomela, U., Kansala, I., Hakkila, J., Mantyjarvi, J.: Context-Studio? Tool for Personalizing Context-Aware Applications in Mobile Terminals. In: Proceedings of 2003 Australasian Computer Human Interaction Conference, OzCHI 2003 (Nokia Research Center), p. 292 (2003)
12. Danado, J., Davies, M., Ricca, P., Fensel, A.: An Authoring Tool for User Generated Mobile Services. In: Berre, A.J., Gómez-Pérez, A., Tutschku, K., Fensel, D. (eds.) FIS 2010. LNCS, vol. 6369, pp. 118–127. Springer, Heidelberg (2010)
13. Cuccurullo, S., Francese, R., Risi, M., Tortora, G.: MicroApps Development on Mobile Phones. In: Costabile, M., Dittrich, Y., Fischer, G., Piccinno, A. (eds.) IS-EUD 2011. LNCS, vol. 6654, pp. 289–294. Springer, Heidelberg (2011)
14. Seifert, J., Pfleging, B., Bahamóndez, E., Hermes, M., Rukzio, E., Schmidt, A.: Mobidev: a tool for creating apps on mobile phones. In: Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI 2011), pp. 109–112. ACM, New York (2011)
15. Holloway, S., Julien, C.: The case for end-user programming of ubiquitous computing environments. In: Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research (FoSER 2010), pp. 167–172. ACM, NY (2010)
16. Danado, J., Paternò, F.: A Prototype for EUD in Touch-based Mobile Devices. In: Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing. IEEE, Innsbruck