Master Thesis

CrowdDesign Touch Enabling complex design tasks on mobile touch devices

Author(s):
Blondin, Alexandre

Publication Date:
2013

Permanent Link:
https://doi.org/10.3929/ethz-a-009781999

Rights / License:
In Copyright - Non-Commercial Use Permitted
CrowdDesign Touch
Enabling complex design tasks on mobile touch devices

Master Thesis

Alexandre Blondin
<ablondin@student.ethz.ch>

Prof. Dr. Moira C. Norrie
Dr. Michael Nebeling

Global Information Systems Group
Institute of Information Systems
Department of Computer Science

24th April 2013
Abstract

In the last decade, we have observed an increased proliferation of mobile touch devices. These devices are often used for browsing content. But how can we enrich the user interaction to allow for content production and complex design tasks on the devices? This thesis presents CrowdDesign Touch, an authoring environment that combines both design-by-example and crowdsourcing to allow complex design tasks to be carried on mobile touch devices.

CrowdDesign Touch builds on the foundation of a previous project: CrowdDesign. This project extends the concept of designing with the crowd to new domains to become a general framework of a mobile authoring environment. Our approach is designed to support authors in composing prototypes of slideshow presentations, mobile applications and mobile websites.

The thesis presents the design and implementation of several prototypes along with experiments that explore the potential of combining design-by-example with crowdsourcing for complex design tasks that are otherwise too difficult to be carried out on a mobile device.
# Contents

1 Introduction 1
   1.1 Aims of Research ........................................... 2
   1.2 Structure of this Thesis ..................................... 3

2 Background 5
   2.1 CrowdDesign .................................................... 5
   2.2 Crowdsourcing ................................................... 7
      2.2.1 The Crowd and Images ...................................... 8
      2.2.2 Text Processing ........................................... 8
      2.2.3 Design Tasks ............................................. 9
   2.3 Design-by-Example ............................................ 9

3 CrowdDesign Touch 11
   3.1 Concept .......................................................... 11
      3.1.1 Slide Editing ................................................ 13
      3.1.2 Mobile Application Design ................................ 15
      3.1.3 Mobile Web Design ......................................... 18
   3.2 Prototyping ...................................................... 18
      3.2.1 First Step: Single Slide Operations ...................... 19
      3.2.2 Second Step: Multiple Slides Operations .................. 21
   3.3 Implementation .................................................. 22
      3.3.1 Initial Prototype — Windows Phone 8 ...................... 23
      3.3.2 Content Extraction ......................................... 24
      3.3.3 Micro Operations .......................................... 25
      3.3.4 Macro Operations ......................................... 28
      3.3.5 Integration with the Crowd ................................. 32

4 Experiments 35
   4.1 Slide Editing - Snippet of Text from a Document .............. 36
      4.1.1 Task .......................................................... 36
      4.1.2 Results ...................................................... 36
   4.2 Slide Editing - Find the Captions of an Image ................ 38
      4.2.1 Task .......................................................... 38
      4.2.2 Results ...................................................... 39
   4.3 Web - jQuery Mobile Tagging .................................... 39
      4.3.1 Task .......................................................... 39
      4.3.2 Results ...................................................... 41
4.4 Web - Recreate a Webpage from an existing Mobile Website . . . . . 42
  4.4.1 Task . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
  4.4.2 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
4.5 Web - Create a Mobile Webpage from a Desktop Website (1) . . . . . 46
  4.5.1 Task . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
  4.5.2 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
4.6 Web - Create a Mobile Webpage from a Desktop Website (2) . . . . . 48
  4.6.1 Task . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48
  4.6.2 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
4.7 Review . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 51
5 Conclusions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
  5.1 Contributions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
  5.2 Limitations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
  5.3 Future Work . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 54
Introduction

Since the introduction of the iPhone in 2007, we have experienced an increased proliferation of mobile touch devices - over 712 million smartphones were shipped globally in 2012, according to press releases\(^1\). With this increasing amount of mobile touch devices, the amount of mobile application and mobile website has been increasing as well. One of the most prominent example is Apple’s App Store that has reached more than 800 thousands mobile applications available in early 2013\(^2\). Where many desktop activities have been successfully adapted to fit mobile environment, the type of task that have been ported are usually made to browse content rather than producing it. For a long time, office suites for mobile allowed the visioning of text, tables and presentation document, but up until recently the modification of content was not allowed. Although now famous solutions exist such as KingSoft Office Suite — which is claimed to be Android most advanced office suites — or Microsoft Office Mobile, the majority of these adaptations appear to have been designed with a desktop computer mindset, i.e. they have not been thought specifically for mobile as they are not optimise for size of the screen nor take advantage of the features of a mobile device — such as touch gesture or device orientation. The major goal of this thesis is how to enrich the users interactions to allow for content production and complex design tasks on a small touchscreen. This thesis has the goal to answer the three following questions:

1. How can the interface be redesigned to enrich users interactions on a mobile device?
2. How can design-by-example be used to help the production of design tasks?
3. How can the crowd be involved in the process of creative design?

\(^1\)https://www.idc.com/getdoc.jsp?containerId=prUS23916413#.UVtyxKuPgzE
\(^2\)http://www.reuters.com/article/2009/04/22/us-column-pluggedin-idUSTRE53L5DK20090422
1.1 Aims of Research

One way to address the challenge of adapting a desktop application to a mobile device would be to simply resize the whole interface to fit the screen size. Using swipe gestures, all the original features could be browsed through as it has been done so far on mobile adaptation of office suites.

Another way to simplify the use of an application is by limiting the amount of functionalities that are made available to the user. In our case, a designed environment that was originally conceived to operate on a desktop computer will have to fit the size of handheld device, which means that not every operation can be displayed — due to the size of the screen. Displaying every functionalities available in powerpoint would overcrowd the screen, subsequently most of these features would not find any use on a mobile touch device. Once the minimal set of features are defined, tasks that are considered too hard to be handled by a mobile device could be outsourced to a crowd of workers. In order to achieve this goal, we present CrowdDesign Touch — an adaptation of CrowdDesign from the work of Nebeling and Norrie [10]. The purpose of the original CrowdDesign project was to provide a design application where new components could be created by the crowd based on existing ones. Our goals differ slightly in that we look at the entire design process and request the crowd to produce content and templates rather than single design elements. Therefore, the aims of the thesis are as follow:

1. Explore different domains that can be performed as a creative design task on mobile touch devices. Compare the related solution that have been developed for both desktop
and mobile touch devices and highlight the features that could be implemented differently in order to offer a better adaptation for mobile devices.

2. Design scenarios where the crowd combined with design-by-example could be involved in order to help the user in his design work. The approach should study which operations are difficult to handle on a mobile touch device and how these complex tasks could be outsourced to a crowdsourcing platform.

3. Carry out crowdsourcing experiments in order to explore the potential of the crowd. The design tasks should tackle problems such as content extraction, production of design templates and tagging of web elements.

1.2 Structure of this Thesis

The thesis starts with Chapter 2 by discussing the background of this project and presenting related work in the field of design-by-example and crowdsourcing. In Chapter 3, we will first describe our process that has driven the overall project and explain our design decisions. Then, we will present the implementation of CrowdDesign Touch. Chapter 4 explores the potential of crowdsourcing through experiments made with CrowdFlower. We will conclude the thesis with Chapter 5 where we will review issues that were encountered in the project, discuss interesting findings and propose new areas that will be explored in future work.
1.2. STRUCTURE OF THIS THESIS
CrowdDesign Touch aims at the adaptation of existing creative tasks from desktop computers to mobile touch devices. In this chapter we will present current streams of work that CrowdDesign Touch is built on. First we will present CrowdDesign (Nebeling and Norrie [12]), which was the main inspiration of this thesis. CrowdDesign is a lightweight developing tool for web information systems that involves a crowd in its design process. Rather than being only a mobile adaptation of the current CrowdDesign, CrowdDesign Touch pushes the whole concept of designing with the crowd further and adapt it specifically for mobile. This project builds on two distinct domain of related work. The first on Crowdsourcing, which represents a good opportunity to tackle challenges that would be to difficult to solve on a mobile touch device. The second, on design-by-example which will provide the user’s work with a starting point.

In this chapter, we will first present CrowdDesign and highlight the major principles to define how CrowdDesign Touch both build on these ideas and also differs from it. We will then cover the background and basics of crowdsourcing and design-by-example by presenting a structured overview of related work on both crowdsourcing and design-by-example.

2.1 CrowdDesign

CrowdDesign presents an approach for lightweight web engineering development involving the crowd in the design process. It supports component-based composition application. The idea is to associate user interface controls with data in a plug-n-play style. In order to do so, CrowdDesign relies on two main component:

- **UI Component**: defines static presentation aspects and dynamic user interface behaviour.
- **Information Component**: consists of the data and the schema which include both structural and behavioural information.
The association between the two is made within a design environment that is embedded directly into the website. The designer can apply operations to the UI Component that allows him to adapt, compose and group interface elements in order to create new advanced components. He can then associate a UI Component with a Data Component to form new widgets. The component-creation process creates two opportunities for crowdsourcing:

1. Crowdsourcing Information Components: allows to automatically generate a data input based on the component schema to be contributed by the crowd.

2. Crowdsourcing UI Component: allows the creation of new Interface Components created by the crowd. The creation can be triggered by a specific request of the user, or based on shared service maintained by a community of users.

CrowdDesign focused mainly on the second scenario of crowdsourcing by providing two approaches. The first approach is based on a community of developers that allows the collection of UI Components to grow. This is done by sharing the components created by users in order to be reused by other designers. It has the benefit to allow non-expert users to benefit from advanced UI Components that they may not have been able to create themselves. The second approach is by having the user making a specific request to a crowd — for example Mechanical Turk — and to let the workers tailor a specific component to suit the user’s needs. This approach have been tested and have generated promising results, which drives our motivation to explore a similar scenario of crowdsourcing, only this time with an editor based on mobile touch technologies.

Similar to CrowdDesign, our project aims at enabling non-expert users to create design interfaces. CrowdDesign was designed to be used on a desktop computer, where CrowdDesign Touch will be aiming on creative tasks on small handheld touch devices. This will introduce new challenges and opportunities of research related to the adaption of main functionalities to fit a smaller screen.

CrowdDesign focused mainly on solving web engineering tasks, in this project we will focus on more general design task by exploring the possibilities offered different domain such as slide editing, application prototyping and web design. This will induce new scenarios where the crowd could be assisting the user. Where CrowdDesign revolved around UI Components and their combination to create new components, CrowdDesign Touch introduces the concept of templates. Rather than having the user combining elements in order to create new ones, we provide the user with a general structure — in the context of slide editing a template could be the layout of a slide, or in web design, a template could be the structure of an article page — once the structure is created, the user can compose views (pages or slides) by introducing the Data Components. This process involves similar scenario where the crowd could be involved. We will explore two domains of creation from the crowd, which are related to the original CrowdDesign:

1. Content Extraction from the crowd: based on an existing source — such as a webpage or a pdf document — the crowd could isolate Data Elements that could later be reuse by the user. This approach would provide the user with a bootstrap of his work.

2. Template Creation: based on an existing example — a slide or a webpage — the crowd could build new layout templates that could later be reuse by users to be filled with Data Elements.
The creation of templates powered by the crowd takes a similar approach than the crowdsourcing of Interface Component proposed by CrowdDesign. It can be both be implemented following two different crowdsourcing models:

- **Sharing and Reuse**: a common library of templates could be built by a community of users. The library could be improved implicitly based on the previous work of a user. The system could automatically extract the structure of a slide or a webpage and add it to its collection for future use.

- **Active Crowdsourcing**: the user could make a specific request to the crowd, providing an example to the crowd, the workers could create a template based on that example.

The second approach of crowdsourcing proposes a combination of crowdsourcing based on micro-tasking platforms and design-by-example. Two domains that are active area of research and that we will present in the following sections.

### 2.2 Crowdsourcing

The term crowdsourcing was originally defined by Jeff Howe [7] and refers to the outsourcing of tasks to a large group of unknown people — rather than outsourcing it to a specific small group of co-workers. Crowdsourcing is an active area of research and has demonstrated a significant success in the market of online labours distribution. Two main trends have emerged from this market, micro-tasking and expert challenges. Platforms such as Innocentive ¹ fall into the category of expert challenges, the idea is basically to outsource difficult tasks — mainly research and development challenges — to a crowd of experts in their domain, thus involving a vast pool of academics and researcher to solve a very specific task. The first challengers to provide a satisfactory solution typically receive a consequent money prize — some challenges rewards its solver up to a million dollars. The counterpart of experts challenges are the micro-tasks platforms such as Amazon Mechanical Turk². The latter refers to platforms that try to reach as many person as possible and take advantage of their idle time [16] in order to solve short-duration tasks in exchange of a small amount of money — usually less than a dollar. CrowdDesign Touch will build mainly on micro-tasking platforms since our goal is to harness as many worker as possible to achieve design tasks at a cheap price. Mechanical Turk has been the subject of many behavioural research that produced many different techniques to optimise the time, the price and the quality of tasks [8, 5, 2], these areas of research are well covered and will therefore not be studied any deeper as part of this thesis.

As described in Chapter 3 and Chapter 4, CrowdDesign Touch will involve the crowd for two major tasks:

1. **Content Extraction** — allows the user to request the crowd in order to provide feedback during the extraction step. Human computation (Quinn and Bederson [15]) has the advantage of being able to find snippets associated to keywords or images.

2. **Template Creation** — allows the user to request the crowd for a new template based on examples.

¹http://www.innocentive.com
²https://www.mturk.com
These two types of tasks are popular crowdsourcing topics in research. In the next sections, we will present an area of work that covers similar challenges.

2.2.1 The Crowd and Images

Image labelling is one of the most popular topic of research that involves workers from the crowd, it has been used widely as one of the most basic experiment to do with the crowd and has induced several research that involve optimising timing and pricing of crowdsourcing tasks [20, 17]. Its popularity is mainly due to the fact that image labelling is one of the task where computers perform badly compared to humans [18], which is one of the main reason behind the popularity of CAPTCHA to certify that an user is indeed a human.

CAPTCHA — ”Completely Automated Public Turing test to tell Computers and Humans Apart” — is a challenge-response test made to ensure that the user of a service is indeed human. It has been used to avoid many types of attacks such as denial of services which affect remote servers or brute-force attacks that break passwords security. One of the most popular example of crowdsourcing for image labelling is reCAPTCHA (Von Ahn et al. [19]), this application is being used over 30 million time a day and allows to digitise small pieces of text.

The goal is to support projects that archive human knowledge by digitising physical books. Because CAPTCHA are being solved over 200 million times a day, it provides an opportunity to use this workforce in the digitisation of text. reCAPTCHA takes benefit of users that have previously ensure their human identity, by successfully passing the CAPTCHA they are great candidate to decipher images of physical texts. The users are challenged a second time and the results they produce are actually used in the process of digitisation. CrowdDesign Touch, will have a similar approach, in the sense that we want the user to start his designing work based on an existing sources. Where images are easy to extract automatically from an existing website or a pdf, texts corresponding to the images — such as caption or description — are a very difficult to map to their images without any human intervention. Our goal is therefore to use the crowd to perform the mapping between images and their corresponding text.

User assisted by a crowd in real time for image-related task is a subject that has been explored in the Crowd in Two Seconds project (Bernstein et al. [4]). The goal is to select the best picture among a collection according to the crowd’s suggestions. Automatic selection of images already exist — many current digital camera takes many pictures when the capture button is pressed, and internal algorithm selects the “best” one — but it seems to be outperformed by humans suggestions. Another aspect of the project, is to have a crowd that helps the user in real time, using an iterative process where the crowd could collaborate in order to converge to an optimal result. Having a crowd running in the background within an authoring environment is the main goal of CrowdDesign Touch as well.

2.2.2 Text Processing

One of the difficult task of writing scientific papers, is to fit length constraint. Many conferences impose a limit of pages for publication. Therefore rephrasing the content in order for it to take less space is a challenge that has been explored with Soylent Bernstein et al. [3]. Soylent proposes a text editor where the user can be supported by the for several tasks:
CHAPTER 2. BACKGROUND

1. “Shortn” — requests the crowd to help the user to shorten a piece of text, using a ruler the user can dynamically visualise the result of the shortening and set the paragraph to the desired length.

2. “Crowdproof” — requests the crowd to help the user to spell-check his text.

3. “The Human Macro” — requests the crowd to help the user in applying a function to an entire piece of text, such as conjugating all verbs into the past tense.

Soylent demonstrated that the crowd can perform very well in the summarising of texts, which is one of the task that will be performed within CrowdDesign Touch. As we have explained, the user should start his work based on an existing source. Because the extraction of text from the source is a difficult process to do automatically, as it would separate the whole the text in chunk that are too big to be used directly, we propose that the crowd should assist the user in this process by summarising part of the text source to fit the UI Components that have been selected by the user.

2.2.3 Design Tasks

Involving the crowd in the process of (re)designing work is an area that have been researched from the production of user studies [1] to the adaptation of webpages in real-time [10, 13, 14]. CrowdAdapt [10] is a good example of how the crowd can be involved in the design of webpages. The variety of devices that allow web browsing has grown significantly in the recent years. The difference between screen sizes makes it very difficult for web developers to optimise their website in order to fit every screens. CrowdAdapt proposes a solution where users can redesign webpages using an online tool. The tool allows operations such as resize and moving within the page itself. The adaptation can be done by the crowd which would then produce versions of the webpages that would fit different size of screen. CrowdAdapt relates to CrowdDesign Touch in the involvement of the crowd in the creation of design templates. In the domain of web design, we propose a scenario where the user would like to start from an existing desktop website and would like to prototype pages designed for mobile. The crowd could be requested to create basic template structure based on existing webpages of the source website, which would involve similar operation than the one proposed in CrowdAdapt. Our approach differs slightly in the implementation, rather than working on the original webpage, we request the crowd to use an UI Builder and to start the mobile webpage from scratch.

2.3 Design-by-Example

Creating template to fasten the design process of the user is one of the main goal of this thesis. Many research projects tackle the same challenges, for example mashup websites have become increasingly popular, tools such as Yahoo Pipes³ can assist the users to easily create new web mashups. As for 2013, programmableweb⁴ counts more than 8’000 APIs and 7’000 mashups. Because of the rapid proliferation of web APIs and the amount of documentation associated, it may be tedious to learn how to use those APIs. Projects such as Koala (Little et al. [9]) or d.mix (Hartmann et al. [6]) aim at using example from the website’s API as

³http://pipes.yahoo.com/pipes/
⁴http://www.programmableweb.com
d.mix offers the most promising approach, the project aims to help developers with basic knowledge of html and JavaScript to rapidly create their web mashups, this is done by leveraging the site-to-service correspondence. The process allows the user to sample elements while he is browsing, the sampled elements can later be added into the website. The sampling process generates the element’s underlying service calls, the code can therefore be easily modified, executed and shared within d.mix’ wiki-based hosting environment. This approach has proven to be effective and has also been applied to create rapid mobile web pages. Based on a desktop website, the user could rapidly sample the page to create its mobile version, which is an interesting opportunity for CrowdDesign Touch. Our process will be similar as we will allow the user to create webpages based on example. However, solutions like d.mix suffer from limitations, not every element can be sampled properly. Because of technical issues, the underlying service is not always extracted as it should, which requires some manual programming to be fixed. We will therefore explore an approach that strictly involves the extraction of design components to allow rapid prototyping rather than extracting components with their services.
According to the aims of research defined in Section 1.1, this chapter presents CrowdDesign Touch — a design editor that allows rapid prototyping of slides, mobile applications and webpages on small-screen mobile devices. CrowdDesign Touch provides the user with additional help based on design-by-example and crowdsourcing. We will start by explaining the approach chosen for CrowdDesign Touch and explains our design decision to highlight the main features of the project. On a second phase the implementation of features and their results will be covered.

3.1 Concept

The main idea of CrowdDesign Touch is to provide a mobile editor that allows the user to do rapid prototyping. As opposed to classical scenarios where mobile touch devices are used to do small adjustment of existing desktop work, we propose an approach where the user handles the whole design process on his mobile and rather than only small parts of it. We aim for the designer to start prototyping directly from his mobile device to provide the general structure of his work. The idea of supporting complex design tasks on a mobile touch device raises new challenges. The size of the screen and the touch input introduce new restrictions, however along with these restrictions, new opportunities arise to explore different solutions. To support all the tasks that are considered too difficult to be achieved with a small touchscreen, we suggest that the project builds on design-by-example and crowdsourcing. These two streams of research offer promising opportunities to tackle the constraints of working with handheld devices.

Conceptually, CrowdDesign Touch follows a four steps process (cf. Figure 3.1):

1. The user starts to work on his mobile, he may suggest a source document\textsuperscript{\small 1} to facilitate the bootstrap of his work. During this step, the user does the ground work by providing

\textsuperscript{\small 1}The source document is typically a pdf or a desktop website. We consider the scenario where the user has done some previous work and want to adapt this work to one of the domain of CrowdDesign Touch.
the general message of his design, this can be done by providing a title, using an exist-
ing template and adding new placeholders with keywords — these placeholder will be
filled by the crowd workers in the following steps.

2. At any stage in the design process, the user can request the crowd to assist him in his
design work. The task should include the current view the user is working on, the
source document (if any) and the description of the task that should be achieved by the
workers.

3. The workers receive the task, typically in the form of a webpage which aggregates all
the elements mentioned in the previous step. The crowd can directly work within the
current slide and provide answers for the user.

4. The user collects the results provided by the crowd, he can visualise the results and can
choose to include them with his current work. He can request the crowd again to refine
his current work or to assist him for another task.

Although we have explained the general process, parts of the overall scenario are still de-
scribed on a very high level — crowdsourcing tasks and templates need to be explained fur-
ther. We have explored three different domains for CrowdDesign Touch. Each differs slightly
in the task that can be crowdsourced and in the concept of design templates. These three domains share common characteristics but have different levels of complexity (cf. Figure 3.2). In the remainder of this thesis, we spend most of our time in the domain of slide editing. It is indeed an easier domain due to its restricted vocabulary. Subsequently due to the similarity of the domains, findings made in the slide editing could easily be generalised to more advanced domains. The design process within every domain follows the same structure. We have three key processes that are present within CrowdDesign Touch:

1. **Content Extraction**: extraction of the content from a source document.

2. **Mobile Design Support**: a context-sensitive interface and support for design templates.

3. **Crowdsourcing Tasks**: a set of tasks that are difficult to achieve directly on the mobile, with the potential to be crowdsourced.

In the next sections, we will make deeper explanations to define each listed features within each domain.

Figure 3.2: The complexity of the vocabulary — the number of elements available in the editor — increases from one domain to another. As we can see slide editing has the most basic vocabulary of all three, whereas web prototyping is much more complex.

### 3.1.1 Slide Editing

Slide editing on mobile touch devices has already been implemented by many companies. The typical scenario usually suggests a businessman on the run that has a last minute modi-
fication to apply on his presentation. Current solutions have therefore been implemented with a lot of options to allow any type of minor edit. CrowdDesign Touch proposes a new scenario: instead of using the mobile device to do small adjustments, the designer could create an entire presentation on his touch device. Our assumption is that the user may want to start the outline of his presentation from anywhere, and therefore would not necessarily be able to use his desktop computer. Our goal is not to fully replace a desktop computer in the process of slide editing. We are in fact trying to cover a different part of the work which involves the creation of early structures of a presentation, the minor details (fonts, colouring etc...) shall be handled on existing solutions that already implements these use-cases.

For the content extraction process, we assume that the designer has a source document which contains valuable texts and images that he would like to use for his presentation. For example, a researcher could start doing his presentation from one of his research papers. The scenario starts with an extraction step which would parse a source document — typically a PDF for the domain of slide editing — and would populate a library of images with the ones contained in the document. The same goes for text that would be stored in smaller parts, thus avoiding the user to retype his work.

In order for the edits to be as simple as possible, we aim to minimise the amount of options that should be available. For slide editing, our design tool does not involve more than two types of objects, texts and images, as they constitute the minimal vocabulary to produce slides. More advanced elements such as SmartArts, shapes or frames are not included in our mobile application.

For the content extraction process, we assume that the designer has a source document which contains valuable texts and images that he would like to use for his presentation. For example, a researcher could start doing his presentation from one of his research papers. The scenario starts with an extraction step which would parse a source document — typically a PDF for the domain of slide editing — and would populate a library of images with the ones contained in the document. The same goes for text that would be stored in smaller parts, thus avoiding the user to retype his work.

In order for the edits to be as simple as possible, we aim to minimise the amount of options that should be available. For slide editing, our design tool does not involve more than two types of objects, texts and images, as they constitute the minimal vocabulary to produce slides. More advanced elements such as SmartArts, shapes or frames are not included in our mobile application.

![Figure 3.3: Typical slides structure that we can see in most presentations.](image)

To fasten the process of creating slide, we rely on templates. In the slide edit world, templates refers to layout. Powerpoint presentations usually involve four types of basic layout (cf. Figure 3.3). This is the basic collection of templates that is originally made available. However, the collection should be enriched based on the previous work of the user. The system should be able to create new templates based on an existing slide. Moreover, we could imagine a community of users that would share their slides. The layouts and their contexts could be then
reused by other designers. Alternatively new layouts could be specifically requested from the crowd by the designer.

For crowdsourcing tasks we have mainly two approaches where the crowd could help the user in the design:

- **Content Extraction**: the crowd could be involved in the process of extraction of content mentioned earlier. Where it is indeed easy to extract images, the extraction of text is much more difficult. Usually pdf extractor allow to use character separator, which is not tailored to our needs, thus producing either single words or big paragraphs as a result of the extraction. One solution would be for the crowd to summarise paragraphs, the user could start his slide by making the basic structure and entering a title, then he could outsource his slide to the crowd along with the paragraph to summarise. Another scenario involve images as certain papers rely on figures for better understanding, an automatic extractor could not link the figures with their corresponding texts. The crowd could again be solicited for the mapping between figures and captions. These two examples of tasks have been implemented as we will see more in detail in Section 3.3.5 and their results were analysed in Chapter 4.

- **Templates Generation**: the user may not be satisfied with the proposed layout templates that are provided. Furthermore, he may not have enough expertise to create his own design template with the authoring environment. To solve this problem, the user may specifically request the workers who will be operating from their desktop computers to create a template for him. To do so, he must provide an example for the crowd that matches his needs (cf. Figure 3.4). The task should include a model that the crowd could use to create the template along with an embedded editor for the crowd to produce such a template. This approach could work by specifically making an open call to a crowdsourcing platform, but it could also benefit from community-based approach. A group of user could make their creations available to other users, thus creating a library of design templates shared within the community.

These two approaches both contribute to create a full slide with the assistance of the crowd. For this project we focused mainly on the micro-tasking approach, as it would provide the necessary incentive to produce results through money prizes. A community-based approach could be feasible as platforms such as Wordpress\(^2\) have shown, but they suffer from a cold-start problem since they rely on a significant community of users to be effective, and would therefore not be recommended as an initial step for our project.

### 3.1.2 Mobile Application Design

Prototyping has a long history in software development, it allows the designers and the developers to get rapid customers feedback thus facilitating the modification of a software project. We propose a scenario where the designer could prototype mobile applications directly on his mobile device. The purpose is to have an immediate feeling of the application in its natural environment. As far as we know, these solutions are available for desktop users — online solutions such as Wireframe\(^3\) or FluidUI\(^4\) — but none of these solutions have yet

\(^2\)http://www.wordpress.com
\(^3\)http://www.wireframe.cc
\(^4\)http://www.fluidui.com
3.1. CONCEPT

Figure 3.4: A task that could be submitted to the crowd in order to produce a new layout template.

been implemented to be used directly on a mobile touch device.
The content extraction process could be made from an existing desktop website. Many web applications have been adapted as native mobile applications. The content extraction could therefore create a library of texts and images based on an existing website.

Because application prototyping’s vocabulary is a superset of slide editing, it should include options related to images and texts. However new components are needed, which includes elements such as buttons, text input or lists. We typically refer to them as “controls”, the design tool should therefore allow the creation and customisation of controls as well as text and images.

Templates in the domain of mobile applications refers to common structures that are found in mobile applications. Similar to the list of templates provided by xCode5 (cf. Figure 3.5), we propose to simplify the user starts to propose such structure as templates. This would include master-detail views, pivot application or simple forms. The toolbar could then adapt itself depending on the template selected for one particular view. Controls may include several elements, therefore, the control options should order the UI elements depending on the template chosen (cf. Figure 3.6). For example, a form view, should have text input and buttons put in priority in the list of controls. Whereas a view with a tabular template should prioritise the elements related to tabs. The templates would therefore include both the structure of a view and the shortcuts related to the view.

The mobile application prototyping offers two opportunities for crowdsourcing:

- **Templates Generation:** similar to the scenario of slide editing, a user could either request the crowd to create a template based on a description and an example.

5https://developer.apple.com/xcode/
CHAPTER 3. CROWDDESIGN TOUCH

Figure 3.5: xCode’s storyboards. A) Simple form View. B) Tabular View. C) Master View. D) Detail View.

Figure 3.6: An example of how templates could combine the structure of a view and the toolbox. The elements that are very often modified within a specific templates appear in the “Shortcut Controls” section, whereas other controls are listed below.

- **UI Elements Generation:** similar to the original CrowdDesign [12], the user could specifically request for design elements that are not found in the default control’s toolbox.
These two applications of crowdsourcing could either be a request to the crowd from a user or could take place within a community of designers. The community would then enrich the library of both templates and controls available within CrowdDesign Touch to allow other designers to reuse their work.

3.1.3 Mobile Web Design

Mobile Web Design is so far the most complex design task that we have imagined that could take place within CrowdDesign Touch. It is a superset of both slide editing and application prototyping, as its vocabulary should also include images, texts and controls. Additionally, web design requires a new type of UI elements, web elements. This represents every element that are only available in a web context, such as widgets or iframes.

The pre-processing step could be based on an existing desktop website. Usually, mobile websites are adaptation of their desktop counterparts. Non-essential elements are typically removed and the rest are redesigned to fit a small touch screen. The extraction process could therefore create a gallery of images and a collection of texts based on the source website. The web domain also benefits from templates as mobile application would. Websites of same categories are often using similar structures — i.e. mobile web articles tend to rely heavily on drop down menus for tables of contents and collapsibles (or lists) for sections of the article. This concept of template is very similar to the mobile application domain and therefore offer similar opportunities for crowdsourcing.

Since web prototyping within CrowdDesign Touch involves a similar pre-processing step than slide editing, and similar templates than application prototyping, the potential of crowdsourcing is the union of the first two domains. We have therefore three scenarios of crowdsourcing:

- **Content Generation:** similar to slide editing, the crowd could be requested for both the summarisation of text and the labelling of images.

- **Templates Generation:** similar to mobile application prototyping, the crowd could be requested for the creation of new templates based on an existing source. This scenario could also take place within a community of designer that would enrich the CrowdDesign Touch library.

- **UI Elements Generation:** similar to mobile application prototyping, the user may have specific needs in term of UI elements that are not covered by the default list of elements of CrowdDesign Touch. The user could therefore specifically request the crowd for it.

Generation of UI Elements is a crowdsourcing experiment that has already been covered in the original CrowdDesign [12], we will therefore focus on the case of crowdsourced design templates. Several experiments to demonstrate the capability of the crowd in the creation of such templates have been conducted within this project and are explained in Chapter 4.

3.2 Prototyping

One of the most challenging parts of the project was to build the CrowdDesign Touch editor. Because of the challenges that are brought from working within a small touch screen and
the vision that includes both crowdsourcing and design-by-example, we needed to carefully design an interface that satisfies our requirements.

To clarify the specifications of our editor, we decided to focus mainly on slide editing. As we have already mentioned, it is the simplest of the three domains that CrowdDesign Touch includes, and the expansion to other domains should be naturally feasible.

During the early stages of the project, we used paper prototyping to explore the main features that the editor should include and how the interface should generally look like. We proceeded iteratively by refining the design step by step based on the drawbacks found and new ideas developed throughout the project.

Our approach followed two major steps. First, we only worked on the edit of a single slide, this would allow us to clarify the operations that should be available and define the general look and feel of the interface. On a second stage, we explored the case of multiple slides, this allowed us to consider options that would affect slides in general and how would they integrate with the single slide editing interface. This methodology allowed us to define two levels of operations:

- **Micro-Operations**: every operation that is made within a single slide. This should include adding elements and modifying them.
- **Macro-Operations**: every operation that is applied on a higher level and affect the whole slide in general, or the whole presentation.

In the two subsections that follow, we detail the two-step process and discuss the results of the paper prototypes.

### 3.2.1 First Step: Single Slide Operations

The first step of paper prototyping resulted in three major iterations. During the first iteration, we created an interface that was very similar to existing solutions — such as KingSoft — the interface contained a lot of options, and different menus. It was not a very original interface, and neither was it effective — both in term of screen optimisation and user-friendliness. We will therefore not present this initial design.

During our second iteration, we decided that we should use side panels to save screen space. Additionally, the side panels should appear only when necessary, and the content of their menus should be sensitive to the context. Therefore, options related to text should appear only if a text is selected and the same goes for images — Figure 3.7 presents the second iteration of the single slide paper prototype\(^6\).

Finally, during the third iteration, we decided to add the possibility to fill the elements with the extracted images and texts from our library. In order to not overwhelm the user with too many options, we decided to separate the editor in two modes that the user could switch to using a horizontal swiping gesture (cf. Figure 3.8):

- **Design Mode**: handles the structure of the slide. The user can add, delete, resize and move elements.

---

\(^6\)To enhance the reading of this document we show clean digitised versions of the paper prototypes, for pictures of the actual physical paper prototypes please refer to the Appendix section.
3.2. PROTOTYPING

Figure 3.7: Paper Prototype — A) The editor in design mode. B) The option panel for text boxes. C) The option panel for images.

Figure 3.8: Paper Prototype — A) The editor in filling mode. B) The option panel to browse images within the gallery. C) The option panel to browse snippets of text from the collection of text extracted.

- **Fill Mode**: handles the content of the elements. The user can select each element and choose from a list which text or image should be the source of the selected element.
Not every design choice were kept during the implementation phase. As will be shown in Section 3.3, the idea of separating the editor in two modes was discarded. The filling and the placing of elements has been implemented in the same view. However, many ideas from the paper prototypes were kept, such as the context-sensitive menus and the side panels to save screen space were implemented.

3.2.2 Second Step: Multiple Slides Operations

During the second phase of paper prototyping we focused mainly on integrating options related to multiple slides. First we needed a way to jump from one slide to another, to do so we introduced a panel that would appear when the user triggered a vertical swipe gesture — from bottom to top. This would make appear a slide navigation menu (cf. Figure 3.9), the user could click on the slide presentation to visualise it. Second we needed a way to add and delete slide, to trigger these operations the user would hold his finger, then the available operations would depend on the finger’s position. If the finger was held on a slide, we suggested the possibility to delete or move the slide. However, if the finger was placed between slides, the application would have suggested to add a new slide at the position of the finger. Macro-level operations, such as templates, themes and crowdsourcing options appeared from a panel above the editor. To trigger the macro-level panel, the user would have to swipe from top to bottom (cf. Figure 3.9).

![Figure 3.9: Paper Prototype — A) The slide navigation. B) The macro-level options.](image)

Having the navigation and the slides operations within the edit mode induced too many options, thus making the interface too overwhelming for the user. During the second iteration, we decided to take a different approach. Rather than using panels for the slide navigation, we decided to handle the slide operations in separate views. From our experience of designing desktop applications, the navigation between slides is usually done with a vertical scrollbar. Our solution was therefore to display the navigation view of slides only when the device was held in portrait and switch to edition when the device was rotated back to landscape (cf. Figure 3.10).

During the implementation phase, the macro-level options — templates, themes and crowdsourcing options — were not kept in the edition mode. As it will be shown in Section 3.3, the macro-level options were moved in the slide navigation view. However, the separation of navigation and edition was implemented exactly the same way as described in the paper prototypes — using the device’s orientation.
3.3 Implementation

In the early stages of the implementation, we decided to first start using Windows Phone 8 technologies. Both Windows 8 and Windows Phone 8 just came out and it was a good
opportunity to test their potential. We were able to create a first prototype very quickly, as it will be presented in the next subsection. However, we envisioned CrowdDesign Touch to be a general framework for design tasks on mobile. Therefore we needed more flexibility for our design environment than the one offered by XAML views. Web technologies offered a good compromise between rapid development and flexibility. Because the coupling between HTML pages and JavaScript is very low, one can very easily modify and manipulate a current view. Moreover, web technologies allowed us to reach a higher number of devices and therefore offered promising opportunities for future work. We decided to use the jQuery Mobile Framework along with jQMultiTouch [11].

3.3.1 Initial Prototype — Windows Phone 8

The Windows Phone 8 prototype, is composed of two views. One handles the template selection whereas the other takes care of the edition of the slide. As can be observed from the prototype’s architecture (cf. Figure 3.11), we rely on two controllers to do the work. One is responsible to handle the selection and the loading of the templates, the other handles every operations that deal with the editor — editing of image boxes and text boxes.

![Diagram](image)

Figure 3.11: The basic architecture of the Windows Phone 8 prototype.

The templates are XAML files stored locally, where the content of a slide is already prepared. The process of loading a template is done in three steps:

1. The user clicks on the interface and selects a template from the list.
2. The Template Selection Controller catches the event, and will parse the content of the corresponding XAML file from the template collection.
3. The Editor View is updated with the content that was previously parsed by the Template Selection Controller.

The Windows prototype included a basic management of templates and allowed few editing capabilities (cf. Figure 3.12).
3.3. IMPLEMENTATION

3.3.2 Content Extraction

Extraction using a PDF file as a source, is a fairly straightforward task. Many libraries exist to perform this operation. We chose to implement a Java applet using iText\(^7\). The applet takes a source as a parameter and then extract both images and chunk of texts to saved them into the local file system.

```java
PdfReader reader;
String fileToExtract = "sample.pdf";
String current = new java.io.File( "." ).getCanonicalPath();
File file = new File(fileToExtract);
reader = new PdfReader(file.getAbsolutePath());
for (int i = 0; i < reader.getXrefSize(); i++) {
    PdfObject pdfobj = reader.getPdfObject(i);
    if (pdfobj == null || !pdfobj.isStream()) {
        continue;
    }
    PdfStream stream = (PdfStream) pdfobj;
    PdfObject pdfsubtype = stream.get(PdfName.SUBTYPE);
    if (pdfsubtype != null && pdfsubtype.toString().equals(
        PdfName.IMAGE.toString())) {
        byte[] img = PdfReader.getStreamBytesRaw((PRStream) stream);
    }
```

\(^7\)http://itextpdf.com/

Figure 3.12: The Windows Phone 8 Prototype — A) The template selection. B) The slide editor. C) The images options.
out = new FileOutputStream(new File(String.format("%05d", i) + ".jpg"));
out.write(img);
out.flush();
out.close();
}
}

Listing 3.1: An example of image extraction using iText.

To make it work online we combined it with PHP’s execute function as it allows php to execute external programs, however this function is considered unsafe by many servers providers, and may not be available in every configuration.

This implementation has been made as a proof of concept, but have not been integrated fully with the CrowdDesign Touch prototype.

Another possibility for helping the user type his text would have been to build a dictionary based specifically from the source documents. However, given the computing power of a mobile device it would have taken a significant amount of time to create a new dictionary every time a source document would have been uploaded. Additionally, modern mobile operating systems usually embed their own dictionary. Having a second dictionary built on top of the native one would have confused the user. We therefore did not implemented this approach.

3.3.3 Micro Operations

As opposed to our paper prototypes (cf. Section 3.2), during the implementation we have decided to merge the “Editing Mode” and the “Filling Mode”. Every operations that are done within a single slide are done within the same mode, we refer to them as Micro Operations. These includes adding elements, modifying their attributes — such as size, position or its source — and handling their deletion.

In order to avoid any conflicting behaviour between jQuery Mobile and jQMultiTouch, the editor was divided in two parts (cf. Figure 3.13):

- CrowdDesign Page: contains the options related to the toolbars, the popups and the overlay panels. These options are made available by the jQuery Mobile framework.

- Editor Page: handles the touchable behaviour of elements provided by the jQMultiTouch toolkit.

The Editor Page was included inside the CrowdDesign Page via an iframe to form a single view for the user. Therefore, the Editor Page represents the content of a slide, where as the CrowdDesign Page is the editor. Elements within the Editor Page are defined by their editable behaviour based on jQMultiTouch touchable features (cf. Listing 3.2). If a new type of element would need to be added — for future expansion of domains — we would need to define a new behaviour for new class of elements.

```javascript
// Text touchable behaviour
var editText = {
  draggable : true,
  resizable : {
    minWidth : 75,
    maxWidth : $(window).width(),
```
As the paper prototypes have shown, one of the key features of CrowdDesign Touch is to have an appropriated interface. The main challenge is to place the most important options on a very limited screen resolution. To achieve this we have implemented a context-sensitive toolbar. Depending on which elements is selected the toolbar should display buttons accordingly. We have therefore created three different settings for the toolbar:

- The general toolbar: allows to add new elements. This toolbar appears when no elements are selected.
- The text toolbar: allows to change the content and the type of the text boxes. This toolbar appears when a text box is selected.
- The image toolbar: allows to delete or change the content of the selected image. This toolbar appears when an image box is selected.
The list of the available options are described in Table 3.14. The system deduces which toolbar to show depending on the user’s behaviour. Every time an element is selected, a jQMutiTouch “TouchDown” event is triggered. Whenever it happens, the class of the element selected will determine the toolbar to show and if the element is deselected the default toolbar comes back (cf Figure 3.15). CrowdDesign Touch could therefore be extended simply by adding a new toolbars for new types of elements.

Every options that allows the modification of an element requires the user to choose from the library. For example, if the user wishes to change the source of an image box, he would have to choose among the elements that are made available from the image library. To save the space on the screen, rather than using a popup window, we used jQuery Mobile overlay panels as it has been suggested during the paper prototyping phase (cf. Figure 3.15.C).

<table>
<thead>
<tr>
<th>Default Toolbar</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Image Box</td>
<td>Add an Image Box to the editor.</td>
</tr>
<tr>
<td>+ Text Box</td>
<td>Add a Text Box to the editor.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Image Toolbar</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Show the gallery of image to be placed in the selected Image Box.</td>
</tr>
<tr>
<td>Delete</td>
<td>Delete the selected image.</td>
</tr>
<tr>
<td>Crowd It</td>
<td>Set the selected Image Box to be filled by the crowd.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Text Toolbar</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Changes the type of the Text Box. 5 Types are available: Title, Subtitle, Standard, Caption and Bullets.</td>
</tr>
<tr>
<td>Source</td>
<td>Shows the collection of text snippets to fill the selected Text Box.</td>
</tr>
<tr>
<td>Edit</td>
<td>Edit the selected Text Box in place.</td>
</tr>
<tr>
<td>Delete</td>
<td>Delete the selected Text Box.</td>
</tr>
<tr>
<td>Crowd It</td>
<td>Set the selected Text Box to be filled by the crowd.</td>
</tr>
</tbody>
</table>

Figure 3.14: Toolbars options.

Another important feature of CrowdDesign Touch, is to do rapid prototyping. Therefore, we did not implement advanced options related to font. The user may not choose precisely the size of his text, neither the font style. The size of the text is handled by the type it is assigned to. We have included five different types that cover most use cases (cf. Table 3.15.D). The colouring is handled by the slide theme, which are handled in the macro-level view (cf. Subsection 3.3.4).
3.3. IMPLEMENTATION

Figure 3.15: The edit mode of a slide— A) The default toolbar. B) The Image Box toolbar. C) Overlay panels that appear when the source button is clicked. D) The type of texts.

3.3.4 Macro Operations

As we have explained in the prototype section, to handle the case of multiple slides, we have separated the editor in two views. One view handles micro operations within the editor, whereas the other handles the navigation between slides and the macro operations. The macro operations refer to the operations that affect either a slide in its whole, or multiple slide at once. The macro mode is activated when the phone is held in portrait. To switch from the macro view to the editor view — where the micro operations take place — the user simply have to rotate the phone in landscape. By doing so, the content of the selected slide is detached and attached into the Editor View. When rotated back to portrait, the content is detached from the editor and reattached to the slide within the macro view. This operation is handled using the Orientation change event provided by jQuery Mobile (cf. Listing 3.3).

```javascript
$(window).on('orientationchange', orientationChangeHandler);

function orientationChangeHandler(event) {
    if (event.orientation == 'portrait' & & (window.location.hash == '#preview')) {
        //The slide selected is copied to the editor, the editor page is then loaded.
        copyToEditor();
    }
    if (event.orientation == 'landscape' & & (window.location.hash == '#content')) {
        //The content of the editor is detached and appended to a div in the slide viewing.
        copyToSlide('.slideSelected');
    }
}
```
In the macro view (cf. Figure 3.16), we have access to two toolbars:

- **A:** The toolbar that handles the advanced operations such as application of templates (layout), themes and crowdsourcing tasks.

- **B:** The toolbar that allows simple slide operations, such as adding, deleting or moving a slide. To use them, the user simply must select a slide and tap on one of the buttons.

We will not detail the simple slide operations, as both their usage and the implementation are trivial. However, advanced operations require to be explained more in detail. The theme options are similar to what we can find in Powerpoint. It allows the user to apply a theme to the whole presentation. As we have mentioned, we did not allow the user to specifically choose the styling of fonts. CrowdDesign Touch being a prototyping tool, we believed that font styling was a too tedious operation to be included. Therefore, to style his
presentation, the user have to use themes from the library\(^8\) (cf. Figure 3.17). To apply the colouring and the new fonts to the whole presentation, we wrapped the slides within a parent div. And depending on the theme chosen we added a new class to the wrapper, each class corresponds to a style defined in a CSS file. To keep the consistency between views, the choices of themes were displayed within an overlay panel as well (cf. Figure 3.17).

\[\text{Figure 3.17: An example of application of themes.}\]

One of the big feature of CrowDesign Touch is to use design-by-example. As we have explained, the templates for slide editing are layouts. Our implementation of layouts allows the user to either create a structure from a empty slide, or to restructure an existing slide according to the selected template (cf. Figure 3.18). To allow this, we parsed the selected slide and tried to map every existing elements that matched the default elements of the template. If an element was found but could not be matched to the template — for example if we had two pictures and we applied the Picture Right Template — the remaining element would have been left in place. Subsequently, if an element should have been part of the template but was not found on the current slide, the element is created anyway with a default content.

Because the slides within a presentation tends to share a similar structure\(^9\). Therefore, we implemented a features that enriches the template library implicitly based on previous slides. To do so, we have created two special options within the template choice:

- **Last Slide**: a layout based on the last slide that has been modified is applied to the selected slide (cf. Figure 3.19).

\(^8\)For more styling options, new themes could be requested from the crowd or generated through a community of users.

\(^9\)It is very common for users to duplicate previous slide to reuse its structure.
• **New From Selection:** a new slide is created with the same structure than the slide currently selected.

To make these features possible, we added a special class to the last slide that has been edited whenever a mode switch occurred — when the device is rotated from landscape to portrait. When the Last Slide button is pressed, we parse the slide that has the last slide class and, for each element found, we recreate an element with a default content in the selected slide. The New From Selection feature is similar. Instead of using the last slide class, we create a new empty slide and, for each element in the selected slide, we create a corresponding element in the newly created slide.

The final advanced option is the Crowd button, this button is part of a larger set of features which will be explained in the next subsection.
3.3. IMPLEMENTATION

3.3.5 Integration with the Crowd

One of the major features of CrowdDesign Touch is the crowdsourcing component. As we have explained, the user should be able to make specific request to the crowd. In our implementation, we focused on the tasks related to content production. To allow the crowdsourcing to work, we decided to integrate CrowdFlower\footnote{http://www.crowdflower.com} to our prototype. CrowdFlower is a service that builds on top of micro-tasking platforms — the most famous one being Amazon Mechanical Turk — because Mechanical Turk is not directly available from Europe, an intermediate platform such as CrowdFlower solves this problem. CrowdFlower uses its own language to build tasks, our goal for CrowdDesign Touch was to be able to generate tasks automatically based on the CrowdFlower markup language. Our implementation involves the following scenario. Whenever a user decide that an element is to difficult to fill, he just press the “Crowd It” button. To implement this scenario we have created three tasks:

- Find Caption: requests the crowd to find the corresponding caption of an image.
- Find Similar Image: requests the crowd to find a similar image than the one currently in the slide.
- Fill The Textbox: requests the crowd to summarise a paragraph in order to fit the elected text box.

These three tasks are common to every domains of CrowdDesign touch. The feature could therefore be used for future extension as part of a whole framework.

Figure 3.19: An application of a template based on the last slide modified.
solved by the crowd. The crowd page, includes the current slide to be crowdsourced and a
document that serves as an example for the workers. The process is done in three steps:

1. The user starts by designing his slide. Whenever an element is too difficult to be filled
within the mobile touch device, he press the “Crowd it” button and chose one of the
available crowdsourcing task (cf. Figure 3.20).

2. When the user has chosen every elements he wanted to be crowdsourced, he press the
Crowd Button from the macro level view.

3. The user specifies a source document that will be processed by the crowd, and just
presses the “Request the crowd” button. From then CrowdDesign Touch will automa-
tically create a CML code that will produce a webpage with a task ready to use (cf.
Figure 3.21).

A more sophisticated implementation would be based directly on Mechanical Turk, this
would allow the process to be integrated automatically through Amazon’s web services. In the
next chapter, we will experiment with CrowdFlower and explore the potential of the crowd-
sourcing scenario we have just described.

Figure 3.20: Integration of the crowd within CrowdDesign Touch— A) The choice of tasks
when the “Crowd It” button is pressed. B) Whenever an element is to be crowded, we add
a text area below it, with the tag of the task. C) Once the Crowd button is pressed, a popup
appears where the user can specify a source document and its type.
3.3. IMPLEMENTATION

Figure 3.21: The page to be included within the CrowdFlower task, this page has been generated based on the slide from Figure 3.20.
During the initial phase of the project, many questions were raised such as:

- What kind of task would it make sense to crowdsource?
- Can the crowd produce design templates?

To explore the potential of the crowd, we propose six experiments based on CrowdDesign Touch scenarios, two for the slide editing domain and four for mobile web design. Throughout this chapter we will present the general goal of the experiments in two steps. First, by providing the reader with a description of the task. Second, by discussing the results of the experiment.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Experiment Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slideshow</td>
<td>Extract snippets of text from a document.</td>
</tr>
<tr>
<td>Slideshow</td>
<td>Find the caption of an image.</td>
</tr>
<tr>
<td>Web Prototype</td>
<td>Tag webpage elements using the jQuery Mobile vocabulary.</td>
</tr>
<tr>
<td>Web Prototype</td>
<td>Recreate an existing website using an embedded web editor.</td>
</tr>
<tr>
<td>Web Prototype</td>
<td>Create a mobile version of an existing website, based on framed elements.</td>
</tr>
<tr>
<td>Web Prototype</td>
<td>Create a mobile version of an existing desktop website.</td>
</tr>
</tbody>
</table>

Figure 4.1: An overview of the experiments that were conducted with the crowd. As we can see they have been ordered based on an increasing degree of complexity.
4.1 Slide Editing - Snippet of Text from a Document

4.1.1 Task

As described in Chapter 3, text extraction is a difficult task to accomplish automatically. Computers are hardly able to separate snippets of text based on their meaning, which therefore results in very low usability for the user because the snippet are often too long to be embedded directly on a slide. The goal of this task is to take benefit of human computation in order to select relevant pieces of text from a document.

For this experiment we have provided the crowd with the first page of a research paper [11] — figure 4.3 — alongside with the following description:

*Below you are provided with a text source, your task is to read the text find the 3 challenges that are expressed. Once you have found the 3 challenges, please write them down in the text boxes below the text source.*

We expected from the crowd to find the 3 following challenges:

• Challenge 1: Many technological differences between touch devices.
• Challenge 2: Different software architectures and implementation methods.
• Challenge 3: Limited support for extensibility.

4.1.2 Results

The experiment was a success, the crowd was able to answer with a high rate of satisfactory answers (cf. Table 4.2). Only one answer was considered unsatisfactory due to its lack of completeness, as we can see below:

• Challenge 1: Technology differences.
• Challenge 2: Different software.
• Challenge 3: Support.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number or judgments collected</td>
<td>5</td>
</tr>
<tr>
<td>Waiting time to get all judgments</td>
<td>3 days</td>
</tr>
<tr>
<td>Average time spent on the task</td>
<td>1m17s</td>
</tr>
<tr>
<td>Level of agreement</td>
<td>60%</td>
</tr>
<tr>
<td>Percentage of “satisfactory” answers</td>
<td>80%</td>
</tr>
<tr>
<td>Price per judgment</td>
<td>20 cents</td>
</tr>
</tbody>
</table>

Figure 4.2: Summary of the first experiment: Slide Editing - Snippet of Text from a Document
Figure 4.3: The first page of the jQMultiTouch research paper [11] as it was presented to the crowd. The workers were expected to find the 3 challenges expressed in the introduction.

This experiment provided promising results as it could be easily integrated within the Crowd-Design Touch crowdsourcing feature and corresponds strongly to one of the scenario described in Chapter 3.
4.2 Slide Editing - Find the Captions of an Image

4.2.1 Task

The second experiment was designed to support another crowdsourcing scenario inspired from CrowdDesign Touch. We wanted the user to be able to select images from his slide and request the crowd to find an appropriate caption to support the image. The task was structured as follows:

*Your task is to find the caption of the image (Image target) from the document below (Document Source).*

For this task we expected the crowd to find the following results:
One of the multi-touch versions we have designed and evaluated for a simple picture tagging application similar to Facebook, here using a two-point tagging interaction.

4.2.2 Results

This task received six judgments in three days. All the worker responded with the same answer:

Figure 6: One of the multi-touch versions we have designed and evaluated for a simple picture tagging application similar to Facebook, here using a two-point tagging interaction.

We did not expect the crowd to take the “Figure 6:” as part of the caption, however it is still a satisfactory result since it fulfil the purpose of helping the user find the content of images captions.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number or judgments collected</td>
<td>6</td>
</tr>
<tr>
<td>Waiting time to get all judgments</td>
<td>3 days</td>
</tr>
<tr>
<td>Average time spent on the task</td>
<td>1m49s</td>
</tr>
<tr>
<td>Level of agreement</td>
<td>100%</td>
</tr>
<tr>
<td>Percentage of “satisfactory” answers</td>
<td>100%</td>
</tr>
<tr>
<td>Price per judgment</td>
<td>20 cents</td>
</tr>
</tbody>
</table>

Figure 4.5: Summary of the second experiment: Slide Editing - Find the Captions of an Image.

As we can see on figure 4.5, the results from the second experiment were very promising, the crowd achieved a perfect score which would have provided good assistance to the user in a real situation. The two experiments within the slide editing domain have showed potential for real-life application of crowdsourcing as part of an authoring environment. However, the potential of the crowd to create design templates based on example is still unknown, this will be test in the next experiments within the mobile web design domain.

4.3 Web - jQuery Mobile Tagging

4.3.1 Task

Tasks that were part of the slide editing domain required no specific knowledge in order to be fulfilled. By contrast, web design is a domain that requires certain area of expertise that may not be known to every worker.

To adapt a desktop website to a mobile device, a designer usually proceed in two steps:
1. Make a selection of the relevant elements of the desktop website.

2. For each element selected, find a mobile placeholder to fit the dimension of mobile touch screen.

The goal of this experiment was to replicate this process and see how the workers of the crowd would perform. To do so, we provided the crowd with a definition of every elements of the jQuery Mobile vocabulary (cf. Figure 4.6). As for the websites that would serve as example, we chose two webpages, one article about crowdsourcing from Wikipedia¹, and the main page of the video game website IGN². To make the task more manageable, we chose beforehand which items from the source webpage should be analysed by the workers by framing them in green figure 4.7. For each items, the worker had to select the jQuery Mobile design element that would fit the best.

Design a mobile web page: by mapping desktop to mobile
In the picture below, you can see different web site elements that are framed and numbered. Your task is to redesign the example web page for mobile by mapping each element to a new mobile widget. Try to follow the example as closely as possible by choosing the elements that best represent each part of the example page. It is sufficient to only reproduce the highlighted elements, and you may also choose to not to include an element in the mobile version.

¹http://www.wikipedia.org
²http://www.ign.com
Figure 4.7: The two webpages that served as a main data for the workers. On the left, an article from wikipedia about crowdsourcing . On the right, the IGN main main page.

4.3.2 Results

This experiments generated complex results due to the density of the jQuery Mobile vocabulary, thus resulting on a significant combinatoric set of possibilities. To evaluate the performance of the crowd we made a list of potential good answers for each elements that were framed from the source webpages. Then, we compared the results of the crowd to our judgment to assess the quality of the work done. We produced two tables that show our judgment compared to the crowd’s. The results of the Wikipedia webpage on table 4.8 and the results based on the IGN webpage are displayed on table 4.9.

![Table 4.8: Results of the third experiment using Wikipedia website to label each elements using the jQuery Mobile vocabulary.](image)

As can observed on figure 4.10, this experiment provided satisfactory results for easy websites such as Wikipedia articles. On the other hand, the IGN main page had rather poor
Figure 4.9: The results of the third experiment using IGN website to label each element using the jQuery Mobile vocabulary.

performances — over 60% of the answers were considered unsatisfactory. The complexity of the IGN website being much bigger than the wikipedia article, this result is to be expected. IGN’s main page is composed of many elements that are not easy to map to jQuery Mobile elements and therefore require a higher knowledge of web design. We deduced from this experiment, that more complex website, such as IGN would need further filtering to harness more experienced worker.

<table>
<thead>
<tr>
<th>Element</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert</td>
<td>Page Header</td>
<td>Select Menu</td>
<td>List View</td>
<td>List View</td>
<td>Radio Buttons</td>
<td>Collapsible List View</td>
<td>Button</td>
<td>Page Footer</td>
<td>Image</td>
<td>List View</td>
</tr>
<tr>
<td>Crowd</td>
<td>Page Header</td>
<td>Select Menu</td>
<td>List View</td>
<td>List View</td>
<td>Collapsible List View</td>
<td>List View</td>
<td>List View</td>
<td>List View</td>
<td>Collapsible</td>
<td>Collapsible</td>
</tr>
<tr>
<td>Grade</td>
<td>66%</td>
<td>33%</td>
<td>33%</td>
<td>33%</td>
<td>0%</td>
<td>66%</td>
<td>33%</td>
<td>100%</td>
<td>33%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Figure 4.10: Summary of the third experiment: Web - Label Webpage Elements

4.4 Web - Recreate a Webpage from an existing Mobile Website

4.4.1 Task

In this experiment, we wanted to test if the crowd could use a web design tool to create the structure of a mobile website. The crowd had to use the jQuery Mobile UI Builder to create a webpage (cf. Figure 4.11). The builder was embedded within the task webpage, and based on two existing mobile websites — again the wikipedia article and the IGN main page (cf. Figure 4.12) — the crowd had to recreate the structure of the example provided.

The task was described using the following instructions:

**Design a mobile web page: based on mobile example**

*Your task is to recreate the example web page shown in the picture using the* [jQuery Mobile](http://www.jquerymobile.com)
editor below. Try to follow the example as closely as possible by choosing the elements that best represent each part of the example page. You can drag and drop new elements from the editors toolbar. For editing an element, simply click on it to view different options on the right. Once you are finished, click on Inspect Code at the bottom of the editor, and copy the result in the HTML Result section.

Figure 4.11: jQuery Mobile Drag and Drop UI Builder.
4.4. WEB - RECREATE A WEBPAGE FROM AN EXISTING MOBILE WEBSITE

Figure 4.12: The mobile webpages that were use for the fourth experiment. On the left, an article from the mobile version of Wikipedia. On the right, the main page of IGN’s mobile website.

4.4.2 Results

In order to evaluate the quality of this task, we had to use a more sophisticated set of criteria. To be considered as satisfactory, we expected the webpage produced by the worker to include all the elements from the model page. The key objective of the web experiments was to assess the capability of the crowd to produce new design templates, we therefore did not expect the workers to retype the text content from the example. The experiment ran for more than a month, but unfortunately we were not able to collect all 6 results, the task was probably too long to accomplish for the amount of money that was awarded — the task took over 11 minutes to complete for a reward of 50 cents (cf. Figure 4.14).

As it can be observed from the crowd output (cf. Figure 4.13), the results for the Wikipedia page are again much better than the one produced based on the IGN website. Out of three answers, only one was considered satisfactory for the IGN webpage, whereas the two pages based on the Wikipedia article were both considered as satisfactory.

Again, we can see that the crowd performs very well with simple webpages, where as it
Figure 4.13: The results produced by the fourth experiment: Recreate a Webpage from an existing Mobile Website. On top the pages based on the wikipedia website, and below the pages based on the IGN website.

appears to be three time harder — based on the percentage of satisfactory results — to produce a good structure based on the IGN website.
4.5 Web - Create a Mobile Webpage from a Desktop Website (1)

4.5.1 Task

This experiment was very similar to the previous one. The workers were again supposed to create the structure of a mobile webpage, only this time instead of a mobile website to serve as an example, the crowd had to use a desktop webpage. In order to simplify the task, the important elements were framed in green — see figure 4.15. Again, the workers were supposed to work with the jQuery Mobile builder (cf. Figure 4.11) and provide us with the resulting html code. The task was formulated as follow:

Design a mobile web page: based on desktop example

Your task is to recreate the highlighted parts of the example web page shown in the picture using the editor below. Try to follow the example as closely as possible by choosing the elements that best represent each part of the example page. It is sufficient to only reproduce the highlighted elements.

You can drag and drop new elements from the editors toolbar. For editing an element, simply click on it to view different options on the right. Once you are finished, click on Inspect Code at the bottom of the editor, and copy the result in the HTML Result section.

4.5.2 Results

For this experiment, a satisfactory result should have included every element from the desktop website. Again, we did not expect the worker to fill the elements with their corresponding content, nonetheless some of the worker did a good job doing it. Overall 6 judgments were collected in a week — figure 4.16. Again, we observed that the Wikipedia website provided much better results than the IGN website, out of three webpages only one was considered satisfactory for the IGN website whereas two fulfilled the requirements for the Wikipedia article — see figure 4.17.
Figure 4.15: The framed webpages that were use for the fourth experiment. On the left, an article from the desktop version of Wikipedia. On the right, the main page of IGN’s desktop website.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number or judgments collected</td>
<td>6</td>
</tr>
<tr>
<td>Waiting time to get all judgments</td>
<td>7 days</td>
</tr>
<tr>
<td>Average time spent on the task</td>
<td>12m24s</td>
</tr>
<tr>
<td>Percentage of “satisfactory” answers (Wikipedia)</td>
<td>66.7%</td>
</tr>
<tr>
<td>Percentage of “satisfactory” answers (IGN)</td>
<td>33.3%</td>
</tr>
<tr>
<td>Price per judgment</td>
<td>50 cents</td>
</tr>
</tbody>
</table>

Figure 4.16: Summary of the fifth experiment: Web - Create a Mobile Webpage from a Desktop Website (Supervised).
Figure 4.17: The results produced by the fifth experiment: Create a Mobile Webpage from a Desktop Website (1). On top, the results from the Wikipedia article. Below, the results from the IGN webpage.

4.6 Web - Create a Mobile Webpage from a Desktop Website (2)

4.6.1 Task

The sixth and last experiment was our most difficult one. It is basically the same experiment that we did in the previous section (Section 4.5) but without the frame on the example desktop webpages to guide workers. The complexity of this task was much higher, because the work-
ers had a wide variety of options when choosing which elements had to be included in the mobile structure.

The task was described as follow:

**Design a mobile web page: based on desktop example**

Your task is to recreate the example web page shown in the picture using the editor below. Try to follow the example as closely as possible by choosing the elements that best represent each part of the example page. You can drag and drop new elements from the editor’s toolbar. For editing an element, simply click on it to view different options on the right. Once you are finished, click on "Inspect Code" at the bottom of the editor, and copy the result in the HTML Result section.

4.6.2 Results

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number or judgments collected</td>
<td>6</td>
</tr>
<tr>
<td>Waiting time to get all judgments</td>
<td>3 days</td>
</tr>
<tr>
<td>Average time spent on the task</td>
<td>5m41s</td>
</tr>
<tr>
<td>Percentage of “satisfactory” answers (Wikipedia)</td>
<td>33.3%</td>
</tr>
<tr>
<td>Percentage of “satisfactory” answers (IGN)</td>
<td>33.3%</td>
</tr>
<tr>
<td>Price per judgment</td>
<td>1 dollar</td>
</tr>
</tbody>
</table>

Figure 4.18: Summary of the last experiment: Web - Create a Mobile Webpage from a Desktop Website (2).

This experiment produced the most mixed results out of all the experiments so far. First, surprisingly both example websites performed with the same level of satisfaction. Second, it is the first experiment that seems to have attracted malicious users. As we can see from the key numbers — figure 4.18 — the task took a significant lower amount of time to complete than its simpler counterpart — over twelve minutes for the previous experiment and less than six minutes for this one. We believe that the high money prize rewarded for this task has drawn more attention to malicious users. This task have seemed to give a first insight of user pattern:

- **Perfectionism:** One worker created the best result that could be attained with the jQuery Mobile builder, the page result not only contained every elements from the model website but also had the content associated to it (cf. Figure 4.19). We did not require the worker to work this hard in order to reach our satisfactory criteria. We suppose that this worker, had probably enjoyed the task and really wanted to do a perfect job.

- **Laziness:** Some users provided some empty results by just fulfilling the minimal required by CrowdFlower, which was done by filling the text area with an empty body element. We had three occurrences of this behaviour (cf. Figure 4.20).
• Carelessness: One worker copied the whole task webpage in the answer text box, instead of only the required html code, thus created a wrong web structure (cf. Figure 5.5). It is a shame, because according to the number of lines of code, the worker seemed to have spent quite some time doing the task.

Figure 4.19: The most impressive result produced by the crowd so far. This page contains a very complete structure for a Wikipedia article alongside with the content of the original article.

Figure 4.20: Three examples of bad results provided by the crowd. The pages are almost empty. On the left, a result based on the IGN website. On the middle and on the right, two results based on the Wikipedia article.
4.7 Review

This set of experiments allowed us to explore into the crowd’s potential for increasingly complex design tasks. Even if the approach was very naive — in the sense that we did not implement any gold standard nor try to optimise the time/money factor — we were able to have a first glimpse of the capabilities of the crowd.

As we have observed in the first two experiments, the crowd can do a fairly good job in assisting a user in his content production — almost every crowdsourcing output for those experiments was satisfactory.

As for template creation, we have basically taken the most complex design tasks that have been included in CrowdDesign Touch, and varied the degree of complexity between the different experiments. We had two examples of websites, one very basic (the Wikipedia article) and one much more complex (the IGN main page). In general, the Wikipedia webpage produced much better results than the IGN main page. However, for every web experiment, the crowd was able to produce at least one (out of three) satisfactory mobile web structures. This is very promising, assuming that mobile web design is one of the hardest design tasks to be carried on a mobile touch device. Therefore, we can assume that the crowd could produce similar — if not better — results for other domains of design tasks.
Conclusions

In this thesis, we presented CrowdDesign Touch — a design framework to support rapid prototyping on mobile touch by combining the potential of both crowdsourcing and design-by-example. CrowdDesign Touch addresses the challenges of working within a small touch screen by involving crowdsourcing in complex design tasks. Together with previous work [12], we regard CrowdDesign Touch as a building block towards new design environments that facilitate complex design tasks by collaborating with the crowd.

5.1 Contributions

In Chapter 3, we have presented an authoring environment that followed a process in four steps. We have shown that this approach is extensible to different domains, thus generalising CrowdDesign Touch as a framework that allows complex design tasks to be created directly from a mobile touch device. Through several iterations of our prototyping work, we have created a new concept of an authoring environment designed specifically for mobile. Furthermore, by combining design-by-example and crowdsourcing, we have tackled tasks that are usually too complex to be carried within the bounds of a small touch screen. CrowdDesign Touch relied heavily on the concept of design-by-example through the use of templates to facilitate the design process of the user. Combined with crowdsourcing, we have discovered new potential for using crowdsourcing in authoring environments. CrowdDesign Touch has defined several scenarios where the crowd could be involved to solve complex design tasks. These scenarios have been explored through experiments with the workers of Mechanical Turk. Through these experiments, we have tested the potential of the crowd and their capacity to solve complex design tasks.

5.2 Limitations

Concerning the investigations around CrowdDesign Touch, several concerns have to be noted:
• When designing the interface of CrowdDesign Touch, we have shown several design alternatives. We chose the most promising design choices based on our own judgment. A user study would have better justified choices as it would have allowed users to provide feedback, thus refining the prototype to a more optimal stage. However, while doing the project, we have identified a set of special requirements for a CrowdDesign Touch user study. The project involves two types of users that interact with each other, we have the designer that operates from the CrowdDesign Touch environment and the crowd that solves the tasks. Because of this setting, one must be cautious when measuring the performances of the prototype.

• The crowdsourcing experiments were carried out with a relatively low amount of judgments. Therefore no clear conclusions can be drawn from the output of the crowd. The experiments did not take into account the possible ways to optimise the quality of the judgments. We did not explore the possible techniques that would have reduced the waiting time or the cost of a design task. Further experiments would also be needed to analyse the incentives of the crowd to solve complex design tasks. These issues have not been covered because they are difficult topics of research of their own and will remain as future work.

• The thesis focused mainly on slide editing as its principal domain. Although we believe that most discoveries that have been made can be extended to the other domains, the difference of complexity would require more research in the domain of mobile application prototyping and mobile web design. As explained earlier, several components — such as templates and crowdsourcing tasks — vary depending on the domain. Therefore, a more complete implementation of CrowdDesign Touch that integrates these domains would be needed to carry out further experiments.

5.3 Future Work

As already mentioned, this thesis has several limitations. However these limitations provide new opportunities for research that would require to be explored further. We plan to drive our research in these particular directions:

• **User Studies:** we plan to challenge the current CrowdDesign Touch authoring environment with different users. The experiment will need to be carried in two steps. First by saving results obtained from previous crowdsourcing experiments, we could make a user study where the user would use the editor and get an offline feedback from the crowd. Doing it this way would avoid to temper the user study with complications that could come from requesting tasks from the crowd in real time. We could therefore focus on the necessary improvements to be made within the design environment. As a second step, an advanced user study could be carried with the tasks being sent in real-time to the crowd. This second experiment will allow the user to assess the quality of the outputs from the crowd in real-life conditions. This two-step process would isolate the different concerns and allow us to focus on the necessary improvements from both the design environment and the crowdsourcing tasks, while minimising the amount of money required.
• **Advanced Crowdsourcing Experiments**: we intend to evaluate the design tasks carried out by the crowd in more depth. For example, we could investigate the possibility of creating a gold standard for design tasks. However, it is a difficult subject since design tasks are open-ended questions, thus making the evaluation of the result very subjective. A way to address this issue would be to submit the output from the crowd to be judged by the crowd itself. Using an iterative process, we could improve on the overall quality of the input. Additional research must be made on the incentives of the crowd in regards to complex design tasks, similar to AutoMan [2], we could test different approaches to optimise the output from a time or a price perspective.

• **Domain Extensions**: we plan to study the domain of mobile application prototyping and mobile web design in more depth. More design tasks must be created to satisfy the differences in complexity that these two domains induce. Moreover, the CrowdDesign Touch prototype will have to include more features related to these two domains. This would allow us to conduct an additional user study tailored specifically for these domains. Other opportunities exist that involve complex design tasks such as desktop publishing or map design\(^1\). Exploring the possibilities offered by other domains and their integration within CrowdDesign Touch should be carried on in our future work. However, it is likely that not all can be supported on mobile touch devices.

---

\(^1\)This could take the form of buildings floor plans, or map creation as it exists in video games.
Bibliography


Appendix: Paper Prototypes

Figure 5.1: On the left the navigation of slides. On the right an early editor interface, as we can see it panels were already there, but they had too many options.
Figure 5.2: A minimalistic interface based on wireframe.cc.
Figure 5.3: The filling mode, used with an image box.
Figure 5.4: An example of template selection.
Figure 5.5: The upper menu that allowed to switch from different modes.