ABSTRACT
Designers of information systems need to address more issues today than ever before because of a recent multi-device context, since in the past few years the possibilities of interaction have significantly grown. To design, test and validate User Interfaces in this complex context is a challenge that cannot be efficiently approached with traditional tools. This work presents a case study of prototyping interfaces on realistic contexts by reporting an experiment conducted in a factory, using a tool that allows designers to produce and test interfaces on multiple devices.

Author Keywords
Prototyping; User Experience; Interaction Design.

INTRODUCTION
Getting the design right in the first time is unlikely to occur for the majority of software projects, because not all the details needed for the implementation are available in the beginning – software design, like any other design, is a problem of wicked nature (the process of designing is the same of understanding) [30].

It is possible to track the causes of failed software designs back to initial phases – if the designers, developers, users and stakeholders do not share a common understanding about what the software should do, who is going to use it, for what purpose, etc. the chances of having a software that people cannot or do not want to use are high. The amount of reworking of flawed designs are unsurprisingly high, and software projects are largely known for surpassing initially estimated budgets [4].

In User Interface (UI) design there are two activities performed during early stages of the development: production and validation of UI’s, and they are currently supported by the combination of many methods and tools involving Sketching as the activity of graphically depicting interfaces of interactive systems; and Prototyping which is the activity of building a limited or scaled version of what will be delivered. Both activities account for designing respectively form and behavior of an interactive system.

There are issues not addressed by previous works in the domain of HCI regarding those two activities, mainly due to the recent phenomenon of the popularity of touch screens and smartphones. At one hand, users now have a myriad of devices to interact, with different sizes and operating systems, factors that have a direct impact on the experience of those users with the designed interfaces. At the other hand, and because of this current multi-platform context, designers need to address more issues than ever before – to design, test and validate UI’s in the current multi-platform context is a challenge that must be addressed.

In this work we present a tool that combines Sketches and Prototypes in different levels of fidelity in order to produce and validate the design of interfaces. We report a user experience study that took place in a factory, by following the work of one designer that was in charge of designing a mobile system for conducting verification procedures on the production line. We have used the system to produce prototypes and collect feedback about the designed solution.

This study builds on a previously work by [29] and improves the initially proposed approach that included a system called GAMBIT – a collaborative tool for User Interface design that allows designers to share pictures and sketches on different devices using a virtual, spatially infinite wall. The system was improved to support prototyping in realistic contexts, as the produced interfaces can be used to construct prototypes for validation. For instance, an interactive prototype can be loaded on a mobile device in order not only to simulate functionalities, but also to reflect real life conditions of use. GAMBIT contributes to expand the set of experience testing tools by offering designers the opportunity to substantially increase the ecological validity of their user studies [13].

This paper is organized as follows: in the next section we present the motivation and a list of requirements for supporting UI design on multiple platforms. In Section 3 we present the tool developed according to the requirements and the related work. In Section 4 we present the study conducted in the factory in order to develop an interactive low-fidelity prototype. Finally, Section 5 concludes.
MOTIVATIONS
The context in which an interactive system will be used is not easily predictable. Nowadays users now have a myriad of devices with which to interact, from smartphones to car-embedded systems, large public displays and home-automation systems. To sketch interfaces and prototype interactions on that diverse context is a challenge, since today the designers need to foresee more contexts of use than ever before.

When designing an interactive system, the designer needs to learn what is the problem to be solved and to design a solution to be used [32]. He/she does that in a cyclic process of producing some artifact (a drawing, a prototype, etc.) and reflecting on it [30]. Currently, the main tools used on that activity are Sketches and Prototypes.

Definitions
Sketches are defined as rapidly executed freehand drawings that are not usually intended as a finished work [9] and it is considered as an important, perhaps necessary, activity for design. The very process of sketching is intrinsic to the design activity, as the designer does not have a pre-defined conscious image of what he/she will draw – this image is in constant construction while he/she is producing lines with a pencil in hand [30].

Prototypes are any artifact that serves the purpose of making a concept (something that is at the designer’s mind) into something concrete, so that other people can use and test it for consistency and eventually provide feedback about the design. A prototype can also be used after validation as a living specification of the functional requirements, as it can be inspected whenever the programmer needs design guidance regarding some functionality. This can save substantial time over the typical development process where programmers sometimes make design decisions on the fly, which may require expensive reworking to fix problems later [28].

While sketches are useful to facilitate discussions on the conceptual level, computer prototypes are useful for discussing operational and interaction issues [18]. Therefore, both aspects (low-fidelity sketches and interactive prototypes) are complementary, and a sort of hybrid approach is considered to be useful since it allows designers to explore alternative solutions for the same problem [21].

By realistic prototyping we mean showing a practical idea of what can be achieved or expected with the artifact being produced (i.e. the interface). As interfaces are necessarily attached to the devices in which they will be presented (i.e. their media), the testing of interface prototypes on the very device it is intended to be used might increase the knowledge of both users and designers about the problem at hand.

Finally, this work considers both designers and engineers of user interfaces on the same Designer role, as the role responsible for providing a solution to a design problem. We refer to Designers and Users in the same way of [34] – Designer is anyone who has the role of creating something (e.g. interaction designers, software developers, etc.) and User is anyone that will use the solution proposed by the designer (e.g. stakeholders, end-users, etc.).

Field Observations
We have observed 2 UI design sessions and conducted 2 interviews with designers in two I.T. companies, in order to devise a method and a software tool that could support their current practices. At the first company, we have observed design sessions that happened over two days during a project for redesigning an information system for a client in the public sector. Three designers and three stakeholders participated in the sessions (two from management, one end-user). The sessions lasted about one and a half hours with two iterations each. At the second company, it was only possible to conduct interviews with the chief designer about the procedures currently in use for designing UI's.

Both companies used Paper Prototyping [33], a classical User Centered Design technique, in order to design interfaces for numerous clients, and the sessions frequently proceeded like depicted in Figure 1 with a cycle of four distinct activities.

1. Drawing: the designers defined the structure of the interface, producing some drawings depicting a user interface in form of screens, roughly one for each “state” of the interface. Very frequently this step was conducted while interviewing the users (clients), with their active participation. Also, designers and users would sometimes be divided into subgroups (one for each “part” of the system e.g. main page and configuration screens).

2. Prototyping: the designers defined the behavior, by assembling the different screens in a logical manner either in a flipbook or in a sequence of screens linked with arrows in order to make the navigation explicit.

3. Sharing/Testing: each sub-group stick their produced prototypes on a wall, arranged in a logical way forming a storyboard. An overall validation happened at that stage, where the navigation flows are “played” or tested – very
often the participants pointed to the drawings following the previously designed navigation, in a sort of “storytelling”.

4. Discussing/Reflecting: Once the design is discussed and the eventual problems are highlighted, the modifications are agreed, then the group would proceed to another iteration of drawing/prototyping/sharing until both parts were satisfied with the results.

Some points were clear after the observations and interviews, and they served as the fundamental concepts for devising the requirements for a software tool to support this process:

A software tool for sketching and prototyping Sketching and prototyping with pen and paper have some important shortcomings, while producing sketches on paper is considered to be easy and fast. iterating (i.e. continuing) those sketches on paper requires effort to duplicate and organize and later register the produced drawings. This is reckoned to be a problem widely documented in the HCI literature [21, 20]. Furthermore, paper prototypes generally require a facilitator, who knows the application thoroughly, to demonstrate or to test the application. Interactivity by the user is somewhat restricted – the user is dependent on the facilitator to respond to the user’s commands to turn cards or advance screens to simulate the flow of the application [28].

Different devices and fidelities Despite the fact the designers used only pen and paper to conduct the two design sessions that we observed, it was reported that they would frequently bring different media according to the users’ needs. For instance, they would bring “pixel-perfect” (high-fidelity) software tools like Axure and Visio, if they needed to design the final look of a system. The users also usually provided pictures of existing interfaces, in case of a redesign and often bring in their own devices or devices that are important for simulating the usage of the system in a real situation. The use of a bar code reader, as seen on top of the table in Figure 1 is an example, and they have also reported that clients would require the system to be designed for a specific models of PDA in which the system should run.

Realistic prototypes The designers also mentioned that it is sometimes difficult to envision a design for a specific platform without seeing how it would look like on it. Also, sketches or interface mockups help only partly to envision how the interaction will really take place. According to the designers, a lot of redesign could be avoided simply by drawing the interface on the very device it is intended to run, for what they use sheets of paper of the device’s size to simulate how the design would look in the real world and also for instance how the device would be held, thus giving them a hint of basic issues like the size of buttons, whether contents is physically accessible by the fingers, etc. In other words, the physical domain would enable designers to make use of the device’s properties that are often lost or ignored using just a paper simulation.

Design vs. Engineering The duality between the design and engineering disciplines is long acknowledged – while designers are trained to think towards solutions, engineers are trained to begin by thinking of constraints; successive constraints are applied until only one solution is valid [26]. According to Rice designers “work essentially from within themselves. They respond to a design challenge by seeking to understand how they respond to the context and essential elements of the problem: their response is essentially subjective.” On the other hand, “The engineer, when faced with a design challenge, will transform it into one that can be tackled objectively. As an example, an engineer might seek to change the problem into an exploration of how to exploit a particular material completely within the context of architecture.” [26].

Therefore, the duality between idea and material, possibilities and constraints, conceptualization and validation is embraced in this work as it is present throughout the process of designing user interfaces of information systems. A designer is worried about user’s requirements and context, and about providing a response to a problem on this specific context – and not so much about the implementation of that response. An engineer is worried about validating that design and on making it feasible, concrete.

Requirements and Related Work
A list of five basic requirements for supporting the four activities previously listed is presented below, together with tools currently available for UI design and prototyping.

Table 1 shows a classification of the available tools according to the requirements. We gave a score of (-) if the tool does not support the requirement in any way or if an extension (i.e. a third party software) is needed in order to achieve the requirement, (+) if the tool does support the requirement and (+++) if the requirement is central to the tool.

<table>
<thead>
<tr>
<th>Tool</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calico</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DENIM</td>
<td>++</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>DAMASK</td>
<td>++</td>
<td>++</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>I-LAND</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TEAMSTORM</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>WallShare</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Sketchify</td>
<td>++</td>
<td>++</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>CrossWeaver</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>InkKlt</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SketchXML</td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UISKEI</td>
<td>++</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Briefs</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Flinks</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>InVision</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Proto.io</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Axure</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Protoshare</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Solidify</td>
<td>++</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Easel</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Justinmind</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1. Tools for UI design classified by the assessed requirements.

R1 Support iterative design of interfaces.

The idea of using unobtrusive tools for aiding design activities is advocated by numerous works in HCI [30, 35, 21] since
it would foster epistemic actions—actions performed to uncover information that is hidden or hard to compute mentally. Software tools that foster epistemic actions are the ones that are not used to yield directly the answer to the problem, but to increase the problem-solver’s understanding of the problem itself and the implications it brings about [34].

Most tools available for UI development nowadays are not focused on UI design, in which designers usually explore different alternatives but on UI modeling as if it is a final product of a process, where designers must attend to formal standards and notations. The flexibility on supporting reflection is an important issue when considering tools for design, and UI design is no exception.

All the tools considered in Table 1 support iterative design, but some of the tools are oriented towards supporting epistemic actions, such as UISKEI [31], Sketchify, DAMASK [20], DENIM [22], and others.

**R2 Support collaboration between designers and users.**

Since design is a problem of wicked nature, it is important to support the collaborative/cooperative approach [27]. Also, the observations reported on Section showed that validation is a crucial element on the UI development life cycle, and this validation is usually done on top of UI prototypes. This validation can occur in what Constantine called “collaborative usability inspections”—systematic, structured reviews of a design, prototype, or working system performed by designers, developers, and users for the purpose of identifying and classifying usability defects [6]. Experienced inspection teams can find 100 or more defects per hour of work.

Most tools are not focused on the collaboration between users and designers, as supporting the three motivations defined by CALICO [21]: Thinking, Talking and Prescriptive. Tools in the intersection between all the three motivations are the ones that can serve as both collaborative and non-collaborative design tools, such as Sketchify, DAMASK [20], DENIM [22] and SketchiXML [8].

**R3 Support different levels of fidelity.**

Section showed that designers frequently use different media according to the users’ needs. Some studies also point out that, for testing design ideas in early stages of design, high-fidelity and computer-based models are not significantly better than low-fidelity [36]. However, the whole spectrum should be covered if we are aiming at supporting the process in a broad way, covering not only designers but also users and other stakeholders.

Most tools only allow one type of fidelity, either low-fidelity like CALICO [21], DENIM [22], DAMASK [20], UISKEI [31], etc. or high-fidelity like AXURE [1], INVISION [17], PROTO.IO [10].

**R4 Support the design of interfaces on multiple platforms.**

A tool that aims at supporting the production of interfaces in the current multi-device context should not force designers to adopt a specific platform. For instance, most of the electronic sketching systems aimed at supporting design and brainstorm sessions proposed in the literature either require large screens [15], custom touch-screen tabletops or equipment that need to be properly calibrated to be deployed, therefore the challenge for current work on UI design is to offer systems that support designers’ activities while being flexible [23, 12].

The design of interfaces in multiple platforms is only supported by PROTO.IO [10] and PROTOSHARE [25]. Other tools like CALICO [21], INKKit [24] and SKETCHI XML [8] are categorized as tools that offers input and output from just one platform (Desktop), thus giving no flexibility for the designers to choose their device of preference.

**R5 Support the prototyping of interfaces on multiple platforms.**

There are numerous references in the HCI literature advocating the focus to be beyond the interface, exploring the interplay among dynamic factors (e.g., urgency, noise) and their impact on usability [7, 16]. As showed in Section, designers would benefit from the possibility of envisioning a design for a specific device if they could test the design on this same device. Also, as designers today need to address more devices and contexts of use than before, due to the usage of devices for a wide range of domains. Therefore, if software tools make possible the use of a prototype in a real device, they should also enable designers to perform user studies and collect data for discussion, potentially increasing the knowledge about the different factors involved on the usage of the interactive system being designed.

Most of the tools do allow interfaces to be tested in more than one platform (i.e. more than one operating system), but only some of them allow the creation of realistic prototypes, like BRIEFS [3], FLINTO [11],-proto.IO [10] and PROTOSHARE [25], this would allow designers to perform user studies in realistic contexts with those prototypes, and also record the user interaction and validate it.

By analyzing the related works we can observe that none of the tools support all the requirements. On the other hand, GAMBIT is positioned as a system that not only can have input and output from many devices, but also from many platforms, ranging from desktops and interactive televisions to smartphones and e-readers. In this sense, devices that are better suited for specific activities can be used if desired (e.g. tablets can be used for pen interaction while large displays can be used for visualization).

**SOFTWARE TOOL**

The GAMBIT tool was constructed based on observations made at design sessions reported in the Field Observations section. The tool was developed to support many devices like tablets, mobile phones, large surfaces, e-paper, etc. and they be organized in many different configurations: for displaying and organizing the sketches (e.g. large displays), producing sketches (e.g. tablets, touch or pen-capable devices) and also testing (with any device). Figure 2 (right) shows one possible configuration—two designers sharing a session with a tablet (bottom right), tabletop (center) and interactive TV (upper right). They are sketching in virtual sheets of paper that can be both produced and shown on the different devices.

We have developed this system aiming at supporting the design of UI’s in the current multi-platform context, in which
both designers and users have many devices to interact, with different capabilities and features. The system is now extended in order to consider prototyping and to support user studies in real (or realistic) contexts in order to enrich the feedback gathered by designers in usability studies.

Figure 2 (Left) shows the main activities supported by the tool and used on the case study. In Step \textit{Produce Interfaces}, the designer can sketch interfaces using the tool in whatever device he/she has chosen (the produced interfaces can also come from external tools or can be pictures taken from interfaces sketched on paper). Step \textit{Produce Navigable Prototypes} shows the interaction map constructed using the interfaces. Step \textit{Test} shows the constructed prototype running on a device (a mobile phone) on a real situation of use, the interaction is recorded for further analysis on Step \textit{Analyze the data}.

The tool allows both \textit{low-fidelity prototyping} – producing an interactive prototype through the use of sketching that does not have itself any functionality (it contains only images), and \textit{high-fidelity experiencing} since the produced prototypes can be embedded on a device and used \textit{in-situ}. In this sense the tool is aiming not only at support UI design, but also to simulate as much as possible of the experience that end users would have with a real system, through the use of embedded prototypes.

The tool addresses all the five requirements, as presented below:

\textbf{R1 Support iterative design of interfaces}: The design activity described in the method is the main concern, since it allows designers to work in many ways, and in flexible environments. Electronic sketching is supported as the main mode of interaction, as it is used to quickly put ideas on an external medium, where they can be discussed, improved and stored for further reference;

\textbf{R2 Support collaboration between designers and users}: It supports group sessions, allowing designers not only to sketch and discuss together, but also to include users in the process. The produced prototypes could be used in real or realistic environments in order to consider contextual information within a user study;

\textbf{R3 Support different levels of fidelity}: Any level of fidelity can be used – low-fidelity such as sketched interfaces or high-fidelity such as interfaces produced with interface builders. Designers might use either \textit{sketching} – in case of devices with touch-screens or pen-based interaction (e.g. WACOM); or by \textit{dragging-and-dropping} interface elements with third-party interface builders (e.g. Balsamiq, Axure, etc.) and then uploading the produced the images; or by \textit{photographing} interfaces produced on paper;

\textbf{R4 Support the design of interfaces on multiple platforms}: Any device can be used to produce interfaces, provided that it has a HTML5 browser and input capabilities (touch or mouse input).

\textbf{R5 Support the prototyping of interfaces on multiple platforms}: The prototyping and testing of systems could take place using the very device it is intended to run. Any device should be used to load prototypes and perform user studies, provided that they have a HTML5 browser. In this sense, it is possible to simulate interfaces and observe interactions on devices that do not have software development kits (SDK) yet available, or if the adopted software technology is yet to be decided.

\textbf{CASE STUDY}

The case study concern an usability testing carried out to test and refine the design of a software system that will be developed to automate a monitoring report based on a series of checklists of equipments in a factory. It took place in a company whose core-business is the production of rust-proof steel sheets for the automobile industry. The designer, a 30 year old woman with experience in usability tests and task analysis, was in charge of designing interfaces for a system...
that would replace a paper-based report used by the workers and management.

**Context**

In order to ensure the quality and the safety during production, workers are requested to fill-in with pen numerous paper-based checklists. They usually consist in a sequence of items where each item represents a check or task that has to be executed by specific workers during their shift.

One of the checklists, called **MONITORING REPORT**, is filled-in daily during a shift by specific workers (to be referred as operators) on the production line. It presents a list of items structured in a table that once filled-in, are handed to the team leader for double-checking and validation. Since the production is continuous, four teams work on the production line in rotating shifts of eight hours. The workforce related to this case study includes overall 32 operators.

**Design Iterations**

The designer was in charge of designing interfaces for a system that would replace the paper-based MONITORING REPORT and deliver them to the development team. The system would have to be mobile, since the operators would carry a device along the production line in order to fill the report, when the production line has a programmed pause for verification. The designer worked closely with the users and developers conducting a series of iterations that began with interviews and analysis of the paper-based MONITORING REPORT and of the context in which it was used.

According to the designer, the work comprised three distinct phases: **Analysis** – where she gathered the initial requirements for the system, developed task models and analyzed the documentation (i.e. the report); **Design** – where the designer developed the screens and prototypes for validation and **Test and Refine** – where the prototypes were tested and validated by the client, and modifications were made in the design accordingly.

**1) Structure Definition**

The designer developed a series of medium-fidelity interfaces using Balsamiq tool [2] that would have to match the device size to be carried by the operators. Figure 3 (left) shows one example of a screen mockup constructed by the designer that contains on the upper part a picture of the equipment to be checked, reference values that should be shown on the equipment, one option OK/Not OK, and two buttons – one for reporting an error on the equipment (the acronym TA is of the vocabulary of the factory and it means *traitement d’anomalie* or *failure treatment*) and one for validating the checklist (thus proceeding to send the report to the chief operator). After producing the interfaces the designer proceeded by inserting them in a GAMBIT session for organizing them in a logical way (Figure 3 right).

**2) Behavior Definition**

In order to create navigable prototypes for discussion, the designer linked the images together as shown in Figure 4 (left), as to simulate a real interaction. For instance, the “radio buttons” OK/Not OK is one form of interaction that could be already simulated in GAMBIT, as the interactors (green rectangles) were placed on top of the options and would lead to another screen. In this sense, the user can make a selection like he/she would do normally, only the system would change one image to another, thus giving the illusion of a realistic interactivity (Figure 4 right).

**3) Test**

The designer then proceeded by conducting brainstorming sessions and showing the screens and prototypes to the client. Figure 5 shows the first brainstorming session we have observed at the factory, with an interactive prototype on a pen-based WACOM device. That session included a senior operator, two people from management and one developer discussing the systems’ functionalities while using a prototype. Those brainstorming sessions were valuable as they served to validate the designer’s work.
4) Reflect: The designer used the feedback gathered on the brainstorming sessions and later on the user studies to improve the prototype, as further detailed. In total, 6 iterations were done to develop the MONITORING REPORT design.

User Study
After some iterations with the management and senior operators, the designer decided to conduct a user study with the end-users, since they would ultimately validate the usability of the designed solution – if they would be able to use the system to conduct the procedure without problems, this part of the system could proceed to development.

Overall 18 workers were recruited to participate in the experiment. Specifically, the group of participants included 15 operators and 3 stakeholders (1 team leader, 1 engineer, and 1 manager), aged 26 to 53 (average age: 42 years), and with an average experience in the organization of 15 years. The participants were scheduled to test the system one at a time and one per hour. Overall, the tests lasted three days (i.e., 6 participants a day).

Figure 6. The experimental setup with four tasks simulating a monitoring procedure.

Figure 6 shows a diagram representing the study setup, we have asked participants to perform four tasks (T1, T2, T3, and T4) each related to a different equipment. The items to be checked were directly extracted from the current paper monitoring report so their checking would be understood and realistic to the participants. The order of items to be checked was also randomized for each subject.

The tests took place in a meeting room located close to the production line where the designer had prepared four 21-inch screens with pictures of the equipment to be checked. Also, a mobile device was prepared with the GAMBIT tool running the prototype. This device is similar to the one that would be used by operators when the system is eventually done.

Figure 7 shows all the screens of the prototype organized in an interaction map highlighting also the current screen being seen on the subject’s device. The subjects were left alone in the room to perform the tasks while the designer stayed in another room observing which screens were navigated while taking notes. The GAMBIT tool recorded the steps taken by each subject (the screens followed by each one of them), together with the time they took. The sessions were also filmed for further reference.

Feedback and Change
At the end of the first day of study some major flaws were identified. Specifically, the six first participants found the validation of checkpoint cumbersome because they had to confirm their answer. Additionally, they experienced difficulties in navigating the system to reach the next item to check. These problems were consistently reported immediately after the test, so the designer proceeded to another iteration, modifying the screens according to the feedback. Despite these modifications the amount of screens remained the same between the two versions of the system.

Figure 8 presents an example of the changes that have been made between prototype1 (i.e., tested by six participants on day 1) and prototype2 (i.e., tested by 12 participants on days 2 and 3). The radio group OK/Not OK and the buttons were replaced by two buttons with a check and a X icon.

User Study Results
The designer conducted a quantitative analysis to compare the users’ performances between prototype1 and prototype2. Both the time (dependent variable: Time in seconds) and the amount of screens looked through (dependent variable: Figure 8. An example of modification made on the prototype based on observations and subject’s feedback. The first two screens (a and b) of the prototype 1 were replaced by one screen (c) in the prototype 2.
Screens) to perform each one of the four tasks T1, T2, T3, and T4 were chosen as the measures of the users’ performances and were extracted from the log files of GAMBIT. Six participants tested prototype1 whereas 12 tested prototype2. Therefore there were two samples per dependent variable: sample1 for prototype1 (N=6x4=24) versus sample2 for prototype2 (N=12x4=48). The results are presented Table 2. Outliers (i.e., measures more than two standard deviations difference from the mean) were removed from samples.

<table>
<thead>
<tr>
<th>Dependent variable</th>
<th>Condition</th>
<th>Sample size</th>
<th>M</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Prototype1</td>
<td>24/23</td>
<td>39.41/35.00</td>
<td>29.88/21.21</td>
</tr>
<tr>
<td></td>
<td>Prototype2</td>
<td>48/45</td>
<td>26.08/19.96</td>
<td>26.77/11.51</td>
</tr>
<tr>
<td>Screens</td>
<td>Prototype1</td>
<td>24/23</td>
<td>9.88/8.61</td>
<td>7.88/5.03</td>
</tr>
<tr>
<td></td>
<td>Prototype2</td>
<td>48/46</td>
<td>3.58/3.20</td>
<td>2.90/2.24</td>
</tr>
</tbody>
</table>

Table 2. User performance summary. M stands for mean, SD for Standard Deviation. Times are expressed in seconds. Results are presented according to the pattern all data/data without outliers.

**Results** An independent-samples t-test was conducted to compare the execution time (Time) in prototype1 and prototype2 conditions. There was a significant difference in the scores for prototype1 (µ=35, σ=21.21) and prototype2 (µ=19.96, σ=11.51) conditions; t(66)=3.93, p<.001. Similarly, an independent-samples t-test was conducted to compare the amount of screens looked through (Screens) in prototype1 and prototype2 conditions. There was a significant difference in the scores for prototype1 (µ=8.61, σ=5.03) and prototype2 (µ=3.2, σ=2.24) conditions; t(67)=6.41, p<.001.

**Interpretation** These results suggest that the prototype does have an effect on both the time required and the amount of screens looked to perform the tasks. This result suggest that the users’ performance increased when using prototype2, since they navigated through less screens to accomplish the same task as with prototype1, therefore the navigation was simplified.

We have also measured the subjects satisfaction by using CSUQ. A five-point rating scale was used instead of 7 (1=totally disagree, 5=totally agree). In total there were two samples to be considered for the statistical analysis: sample1 for prototype1 (N=6x18=108) versus sample2 for prototype2 (N=12x18=216). Results per prototype are presented Figure 9.

**Results** An independent-samples t-test was conducted to compare the user satisfaction (all item score) in prototype1 and prototype2 conditions. There was a significant difference in the scores for prototype1 (µ=4.06, σ=0.78) and prototype2 (µ=4.37, σ=0.65) conditions; t(322)=-4.54, p<.001. Similarly, an independent-samples t-test was conducted to compare SYSUSE, INFOQUAL and INTERQUAL scores in prototype1 and prototype2 conditions. The differences were all significant (p<.05), especially in the INFOQUAL scores for prototype1 (µ=3.81, σ=0.79) and prototype2 (µ=4.36, σ=0.64) conditions; t(106)=-4.49, p<.001.

**Interpretation** These results suggest that the prototype does have an effect on user satisfaction. This result suggest that when users use prototype2, their satisfaction increase in particular regarding the information quality. In addition, these results are very consistent with the subjective general comments that were made during the debriefings. Users testing prototype1 (i.e., group1) mainly focused on their validation and navigation concerns as described previously in “Feedback and Change”, whereas users testing prototype2 (i.e., group2) did not even mentioned them (see INTERQUAL scores).

**DESIGNER’S FEEDBACK**

The work of the designer was difficult in this particular study because of the context in which it took place – at the factory, the production is continuous and requires five workers to run almost permanently. Workers are under time pressure and need to attend primarily to work tasks, so it was very difficult to recruit them to attend to sessions. Furthermore, the work takes place in an unfriendly environment, with noise, heat or cold, risks of accidents, etc.

The designer reported to having received significant feedback about the solution she had designed, during the brainstorming sessions, but most importantly we have observed that the discussion that took place on those sessions was about the problem of conducting the monitoring report on the production line, as users started to discuss about the procedure itself as soon as they interacted with the prototype.

The participants of the brainstorming sessions focused almost exclusively on the experience in the factory rather than on the system itself: operational organization (e.g., where and when to put the mobile device in charge or to store it), workday organization (e.g., how the system would impact a workday), or even foreseen uses (e.g., how such a system could be used...
to support other tasks in the production line). A different understanding about how the business itself operates was also evident between operators, management and developers, and the prototype served as an artifact of discussion to make clear the distinct views to all the participants.

Table 3 shows the total cost expressed in man-hours of the project according to the designer. We can observe that there is a separation between activities In the Lab (i.e. the time taken for internal work like designing, prototyping, planning and documenting) and In the Wild (i.e. testing and evaluating the design at the company). Results show that analysis, design and test phases respectively cost 78 man-hours (36% of the overall time), 54 man-hours (25% of the overall time) and 84 man-hours (39% of the overall time).

<table>
<thead>
<tr>
<th>Step in UE life-cycle</th>
<th>Deliverable</th>
<th>Wild</th>
<th>Lab</th>
<th>Total</th>
<th>Per phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Report interviews</td>
<td>40</td>
<td>10</td>
<td>50</td>
<td>78</td>
</tr>
<tr>
<td></td>
<td>Task model</td>
<td>8</td>
<td>20</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td>Paper mockups</td>
<td>8</td>
<td>8</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Balsamiq mockups</td>
<td>38</td>
<td>38</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>Test and refine</td>
<td>User testing</td>
<td>48</td>
<td>16</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data analysis</td>
<td>10</td>
<td>10</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>Final UI design</td>
<td>10</td>
<td>10</td>
<td>20</td>
<td>84</td>
</tr>
</tbody>
</table>

Table 3. Cost table (expressed in hours) for the three initial iterations of the project, in respect to three phases (Analysis, Design and Test).

The low cost of both design and test phases (i.e. only about 7 and 9 man-days respectively) is aligned with observations reported in the literature: delivering low-fidelity prototypes is less expensive than delivering high-fidelity prototypes and can still disclose a similar number of critical problems as prototyping techniques with higher levels of fidelity [5]. Additionally, engineering the modifications between prototype1 and prototype2 required 6.5 man-hours, i.e., 3 man-hours to produce the Balsamiq mockups and 3.5 man-hours to modify the interaction map accordingly. In other words, it only costs about 3% of the overall engineering time to refine the prototype which significantly increased both user performance and user satisfaction.

Finally, the designer reported that the tool aided her work, as GAMBIT supports the design and testing of prototypes on a realistic environment, approximating very closely the interaction with a real system. Eventually, user behavior, attitudes and emotions can be investigated in natural context of use and problems can be identified before the final software is produced.

CONCLUSION

People who have developed a system think differently about its use, and this is the main reason why, as Gould pointed out in [14], “reviewing” or “demonstrating” a prototype system for typical users and getting their reaction can result in misleading conclusions. Instead, he suggests, users should be given simple tasks to carry out, and their performance, thoughts, and attitudes should be recorded and analyzed.

We presented a tool and a method for supporting Sketching and Prototyping of User Interfaces, by allowing the production of interactive solutions on multiple devices and their testing in real situations. We have followed the work of a designer on the task of prototyping and testing interfaces with GAMBIT.

The main advantage of using the tool proposed in this work is that it enables designers to know in advance if a solution will work without investing time and resources on making this solution concrete (i.e. coding). Furthermore, the possibility to test prototypes on a real device gives designers the chance to make informed decisions about the solutions they are designing. For instance, a designer might choose one type of device or another based on the screen size, if controls and buttons are better reached by users in one device than in another. Other properties such as environment’s noise and light level might also have an impact on such decisions.

The case study demonstrates the contributions advocated in this paper. On the one hand, prototyping and testing have been successfully supported by GAMBIT and its functionalities have proven efficient means to improve the productivity of the designer. Although the user experiment was conducted with low fidelity prototypes, GAMBIT contributed to approximate rather closely the user’s natural context. In particular, the system was tested by the targeted end users on the very device the system is intended to run, and without any pretest training and without any assistance by the experimenter. This makes GAMBIT particularly convenient and reliable to support UX design.

On the other hand, test and refine prototypes are key-activities in UI production and designers cannot afford to invest important resources in programming. Consequently, it is important that non-developers are able to efficiently and autonomously produce cost-effective prototypes. GAMBIT enables designers to substitute programming by simply building an interaction map between the screens of a lo-fi prototype. It is also important to provide designers with efficient means to analyze the interaction, and the tool provides them with the possibility of monitoring users’ actions both in real time and post-experiment.

REFERENCES


