From Research Laboratory to the Market: the Experience of *TypeInBraille*

Sergio Mascetti
University of Milan and
EveryWare Technologies
mascetti@dico.unimi.it

Cristian Bernareggi
University of Milan and
EveryWare Technologies
bernareggi@dsi.unimi.it

Andrea Gerino
University of Milan and
EveryWare Technologies
gerino@dico.unimi.it

1. INTRODUCTION

While the research for assistive technologies is an active field since the middle '70, new challenges and opportunities arose in the last years, due to the diffusion of accessible mobile and pervasive devices. Among the others, there are new challenges in the design of interaction paradigms for these devices that are characterized by small physical dimensions and different interaction hardware (e.g., input through touchscreen versus keyboard and mouse). Concerning the opportunities, the pervasiveness and portability of these devices allow the users to benefit from assistive technologies each time of the day. The research conducted in the last years leads to the design of brilliant solutions, both to address some of the challenges and to take advantages from the opportunities.

Despite a significant research effort was devoted to develop a number of working prototypes, only a few of them became actual products from which final users could take advantage. This is due to two intertwined reasons: first, the legitimate goal of researchers in this field is to devise solutions, rather than develop them for distribution to end users. The second reason is that a great effort is required to deploy a final product starting from a research prototype and that this process requires economical investment as well as some competences that are not always available in a typical research team, like technologists for the implementation of the application and its maintenance, experts in communication, to enhance the diffusion of the application, and translators to localize the application in different languages.

In this contribution we want to share the experience of our team that, starting from the scientific research conducted in the university laboratory "EveryWare Lab" [1], develops applications and commercializes them through the spin-off called "EveryWare Technologies" [2]. In particular, we will focus on an application called *TypeInBraille*, that allows visually impaired persons to type on touchscreen devices using Braille encoding [3, 4]. We first briefly describe the research and development steps that the application has undergone (Section 2) and then we discuss some problems that we encountered and lessons that we learned from this fascinating experience (Section 3).

[1] `http://everywarelab.dico.unimi.it`
[2] `http://www.everywaretechnologies.com`

2. RESEARCH AND DEVELOPMENT

Design methodology. *TypeInBraille* was designed and developed according to a participatory user-centered design methodology, following the star life cycle model of an interactive system [2]. In the star life cycle, the results of each design and development stage must be evaluated by end users and by the designers before passing to another stage (e.g., evaluation of the software prototype may lead to revise requirements specification). Hence, the product gradually evolves thanks to the continuous interaction between users and designers. It is worth observing that our team includes a blind designer proficient in Braille and expert user of iOS devices that was involved both in the design and the evaluation of *TypeInBraille* since its early stages. A consequence of this methodology is that the design and development of the product do not end with its first release, since it evolves according to new user needs emerging from evaluation sessions and product use.

Identification of a typing technique. In order to test different typing techniques we implemented an iPhone application to record the user gestures so that the resulting output could be interpreted off-line with the help of a semi-automatic application. During this stage we involved two blind users in the testing of a large number of different input techniques. The advantages of the semi-automatic interpretation of the result resides in its flexibility and easiness to adapt to different input strategies hence making it possible to experiment very different solutions with a small coding effort. At the end of this phase, thanks to the users' feedback, we identified a set of three possible typing techniques.

The proof of concept. In order to decide which technique to adopt, we implemented a proof-of-concept application for each of them. The only functionality provided by each application was typing. The gestures interpretation was performed "on-line" so that the applications could give audio feedback to the user. A group of 7 blind users evaluated these applications. Users were proficient in writing and reading Braille and well acquainted with the on screen QW-

ERTY keyboard. The aim was to evaluate the typing speed and error rate comparing the three solutions both among themselves and with the default iPhone QWERTY. The analysis of the result highlighted that one of the proposed solutions outperformed the other two as well as the default QWERTY keyboard. The idea of this solution, which is at the basis of the *TypeInBraille* application, is to enable users to type a character on the touchscreen through its Braille representation by inserting, one by one, the three rows that compose each Braille character (see Figure 1). Since each

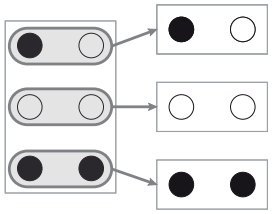
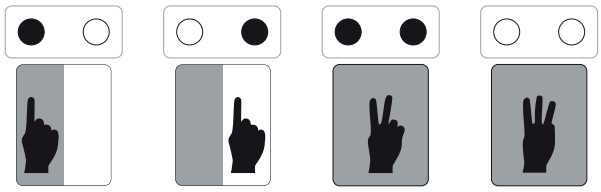


Figure 1: Subdivision of each Braille character into three pairs of dots.

row is composed by two dots, four gestures are defined to represent each possible combination of a row. These gestures are based on a subdivision of the screen in two parts: one tap on the left part corresponds to the left dot raised and the right dot flat, one tap on the right part corresponds to the left dot flat and the right dot raised, a two-finger tap corresponds to two raised dots and a three-finger tap corresponds to two flat dots (see Figure 2).



(a) Left dot (b) Right dot (c) Both dots (d) No dot

Figure 2: Gestures defined to enter a pair of dots.

From users tests we also collected useful feedbacks that lead to the implementation of new features, including: a vibrational effect upon symbol insertion and additional gestures to speed up the typing of frequent symbols, like the blank space or the new line.

The application prototype. After identifying the most promising typing solution we developed a prototype application that included additional features, like text navigation and selection. We conducted two tests. The former was conducted by a blind user for a long period (about one month) and was intended to estimate the typing performances of a trained user. As a result we observed that the trained user was able to type about 2.5 times faster and with a much higher accuracy with respect to the on-screen QWERTY. The latter test involved 9 blind users for about 2 hours each in two real-world contexts: at the desk and on a tramcar. In each context, *TypeInBraille* was compared with the on-screen QWERTY keyboard according to typing speed and accuracy. The results highlight that *TypeInBraille* outperforms the on-screen QWERTY in both contexts and that the difference between the two techniques is even more remarkable in the tramcar scenario since *TypeInBraille* can also be used with partially audible feedbacks [4]. Also in this case, we collected valuable user feedback that allowed us to improve the prototype. For example the original application, according to the iPhone UI best practices, adapted itself to the device orientation. However, as many users pointed out, while using the application in an uncomfortable environment like the tramcar, it happened relatively frequently that the device was accidentally rotated, hence resulting in an hindrance for the user. For this reason, we decided to keep the rotation of the application fixed in portrait mode.

The application deploy and improvement. The deploy of *TypeInBraille* on the AppStore was delayed due to limitations with iOS in its former versions. The problem was that it was not possible for an application to directly capture screen touch events when VoiceOver was enabled. Consequently, it was not possible to detect the gestures and, at the same time, to give speech feedback via VoiceOver. The prototype application had overcame this problem by using undocumented APIs that, however, cannot be used in applications deployed on the AppStore. This limitation has been removed since iOS version 5.0 hence making it possible to deploy the first public version of the app. Since its publication, at the end of October 2011, three major updates of the application were released introducing new features and customizations.

3. PROBLEMS AND LESSONS

The research, design, development and deploy activities related to *TypeInBraille* posed many different problems. Addressing these problems, we learned many valuable lessons that we believe can be applied in the research and development of other pervasive assistive technologies.

3.1 The importance of customization

The first set of problems we run into concerns the crucial role of customization for a pervasive assistive application as *TypeInBraille*. We can identify two kinds of customizations: those based on the user profile and those based on the context of use. The former category includes language localization as well as other special needs by particular classes of users. Concerning localization, in addition to translating the application messages into several languages, a not neglectible effort was required to localize the Braille code for each locale. Note that there exist at least one Braille code for each language and, for some languages (e.g., English), there are different character mappings in different countries (e.g., US and United Kingdom). Moreover, two Braille representations are in use in English: contracted and uncontracted. Furthermore, some non standard Braille representations of certain characters are widely used by some Braille readers (e.g., by those who read on a Braille display).

Concerning the customization for particular classes of users, *TypeInBraille* has been extended with many features sug-

gested by users themselves. For example, many comments from users required a feature to show the Braille dots and entered text in order to facilitate the communication between visually impaired and sighted people. Figure 3 is a screenshot of the application showing the entered text and the dots composing the character being entered. Another feature ad-

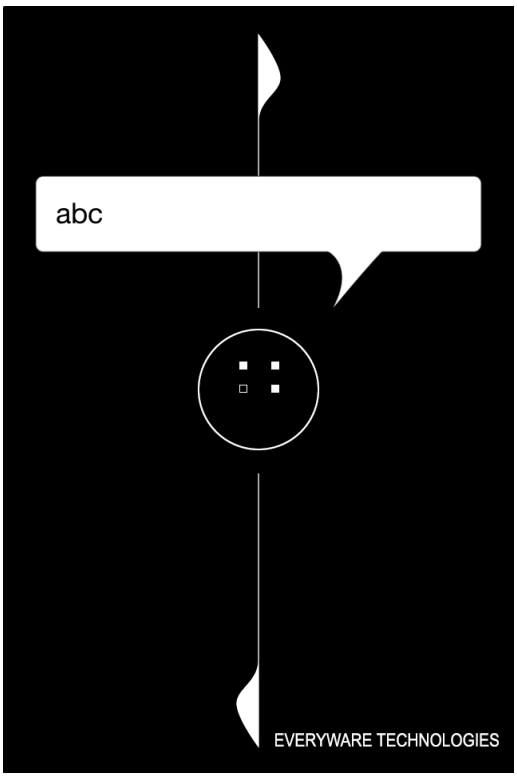


Figure 3: Screenshot of *TypeInBraille* while entering the letter 'd'.

dresses the reading needs of partially sighted people, that can benefit from receiving magnified graphical feedbacks. Similarly, we received requests from users with multiple disabilities, for example related to the hindrances that may emerge in the use of the "rotor" gesture for partially motor impaired.

The latter category of customizations is related to the context of use. For example, users reported different needs while writing in a quiet or noisy environment. Indeed, in the former case the vibration effect is disturbing, while in the latter case it is helpful to notify the user that a character is entered. Similarly, on one side, some users requested an input technique that is less mentally and physically demanding (e.g., a "one finger" input technique that requires six gestures to enter each Braille cell). This is intended to be used in uncomfortable environments, in which, for example, one hand only can be used to type. On the other side, other users asked for a more requiring technique, like BrailleTouch [1], that is designed to be used with two hands and is intuitively more suitable in a comfortable environment.

From these various requests we learned that, although we designed the application mainly for blind people, the actual set of users is much broader and it is necessary to consider other different user needs. Consequently, users with dis-homogeneous characteristics should be employed in the tests. Moreover, we designed the input technique with the idea that it could fit many different contexts. However, there is not a single technique that best suits every situation. For this reason, it is necessary to give the user the opportunity to choose the preferred typing technique depending on the context. Clearly, the mental effort to switch among the techniques should be limited. In this view, the underlying adoption of Braille is a common feature in potential different typing techniques and can leverage the user from learning many different encodings.

3.2 The interaction with users

The maintenance and improvement activity after the deploy was driven by user feedbacks collected from two main sources: direct communication with the support team and discussions among the users on forums and social networks. In order to increase the number of user feedbacks, a very simple but effective functionality is to allow the users to write to the support team directly from inside the application.

Overall, after four months from its release, there are about 50 threads of discussion about *TypeInBraille* in English and we received about 100 messages from distinct persons. Consequently, a great number of different remarks and suggestions was collected and analyzed by the design team, leading to the improvement of the application. Among the most requested updates, users asked for localization in different languages, the implementation of new functionalities (e.g., direct publishing on social networks) and the adaptation of the application to special user needs and contexts.

The users have also proved to be proactive in the creation of new content related to the application. Indeed, many users in the discussion forums provided descriptions of the application, comparison with other solutions and support for other users. Among the other, some users shared screencasts or audio-descriptions of the application, while other users proposed to participate in the process of application localization, by providing the localized Braille tables and by translating the application messages, the description and the help. The difficulty in this process was that it was managed manually, by using one-to-one communication media, mainly email hence resulting to be a time-consuming task.

Another problem that we solved with the help of the users' community is the dissemination of information about the application. Since the resources for an advertising campaign were very limited, we started publishing information on our website, on some social networks and in some forums. We observed that the users' contribution in the diffusion of the information is fundamental. Indeed, users are active in reporting the information on other forums and local discussion groups, also translating the information when necessary.

From this experience, we realized that the community of visually impaired persons is very active and ready to provide support, feedback and cooperation. While, on one side these contributions are of paramount importance in the improvement of the application, on the other side, the management of the interaction activities can be time consuming and requires that a member of the team devotes part of his/her

time to this activity. For these reasons, we are planning to implement a platform to facilitate the collection of contributions by the users. For example, one functionality of this platform will be to create the application localization in a collaborative manner by presenting the English version of the text to the users and allowing them to contribute with the translation in their own language.

3.3 The importance of system support for assistive technologies

The existence of accessible devices and operating systems is a necessary condition for the development of assistive applications. However, this condition is not sufficient in many cases. Indeed, it is necessary that the operating system is sufficiently flexible to make the OS assistive functionalities available at application level. For example, *TypeInBraille* relies on the operating system method that "reads" custom messages via VoiceOver. This is very useful to read both the single characters while they are inserted, and the entire words and sentences when required by the user. It is clear that the possibility to call this method from a custom application significantly simplifies the application development and also makes the application more consistent with the operating system and the other applications.

Note that there is a continuous improvement of the API related to accessibility. On one side, this enhances the development of assistive applications. For example, *TypeInBraille* was deployed after the introduction, in iOS 5.0, of the speech method in the accessibility API. On the other side, the frequent updates also pose some maintainability problems. For example, in the recent update from iOS 5.0 to iOS 5.1, some accessibility-related APIs were changed, introducing a bug in *TypeInBraille*. Nevertheless, the improvement of the accessibility APIs is a necessary process, that is dictated by the new necessities arising in the research and development of accessibility applications. Indeed, *TypeInBraille* would take advantage from some APIs that, however, have not been implemented yet. One example is related to an issue with VoiceOver's focus mechanism: the OS forwards touch events to the application only when they occur in a focused view. However, the focus can only be changed by the user and there is no API that makes it possible to change it from code. As a consequence, when a view is presented, the first touch is always used to acquire the view focus and hence cannot be used by the application itself. This may be inconvenient, like in the case of *TypeInBraille*.

The lesson that we learned is that the mobile OSs are frequently updated and many new release introduce new accessibility related APIs. These improvements are rarely presented as important new feature, and it is necessary to go trough the "change log" to discover them. This process is time consuming but it can unveil new enabling features.

4. CONCLUSIONS AND FUTURE WORK

Users' contribution is a central aspect in the design of assistive technologies. While many useful comments and performance data can be collected while testing the application prototypes, the number of users that take part in the tests is generally limited to few units, or tens. Vice versa, when applications are distributed as final products through the market, the number of users that can be reached is orders of magnitude higher, even with a small marketing campaign.

As we have motivated in this contribution, a significant effort is required to transform a prototype into a final product that can be distributed on the market. However, this effort is justified by two major results that can be reached. The former is that many users in the world can benefit from the results of the research. The latter is that, as we have experienced, users are eager to be involved in the improvement of the solution, hence they are proactive in providing feedback, highlighting problems and proposing new features.

From the scientific point of view, deploying application on the market enables the collection of experimental data from a much wider audience, in many different contexts of use as well as from users with different disabilities and needs. For these reasons, we are planning to introduce new features in *TypeInBraille* so that the application will automatically collect performance-related information, like, for example, the typing speed and the number of undo/delete operations. For the same reason, in order to perform a comparison among different typing techniques, it will be necessary to implement other typing solutions to be included in the same application so that users can be able to rapidly switch among the typing techniques according to the context and their needs.

5. REFERENCES

[1] Brian Frey, Caleb Southern, and Mario Romero. Brailletouch: mobile texting for the visually impaired. In *Proceedings of the 6th International Conference on Universal Access in Human-Computer Interaction: Context Diversity*, UAHCI'11. Springer-Verlag, 2011.

[2] Deborah Hix and H. Rex Hartson. *Developing user interfaces: ensuring usability through product & process.* John Wiley & Sons, Inc., New York, NY, USA, 1993.

[3] Sergio Mascetti, Cristian Bernareggi, and Matteo Belotti. Typeinbraille: a braille-based typing application for touchscreen devices. In *In Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM, 2011.

[4] Sergio Mascetti, Cristian Bernareggi, and Matteo Belotti. Typeinbraille: Quick eyes-free typing on smartphones. In *Proceedings of the 13th International Conference on Computers Helping People with Special Needs.* Springer, LNCS, 2012, to appear.