From Desktop to Touchless Interfaces: A Model Based Approach

Franca Garzotto, Mirko Gelsomini, Roberto Mangano, Luigi Oliveto, Matteo Valoriani
Politecnico di Milano
Via Ponzio 34/5 20133, Milano, Italy
{franca.garzotto, mirko.gelsomini, luigi.oliveto, matteo.valoriani}@polimi.it, nickelnext@gmail.com

ABSTRACT
With the increasingly low cost of motion-sensing technology, touchless interactive interfaces may become a new ingredient in the evolution of content intensive web applications from single-platform (desktop) to multi-platforms use. While the migration from desktop to mobile devices has been widely studied, there is limited understanding on how to include touchless interfaces in this "going multi-channel" evolution. The paper focuses on the design issues that are induced by this process. We propose a model-based design approach that supports information reuse and exploits a systematic mapping from content structures to interaction tasks and touchless gestures. We then describe a case study in the cultural heritage domain to exemplify our method.

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation]: Multimedia Systems, User Interfaces

General Terms
Design, Experimentation, Human Factors

Keywords
Touchless interaction, motion-based interaction, design, model-based approach, Kinect, large display, web engineering

1. INTRODUCTION
Touchless Motion-based Interaction (TMI for short) enables users to control digital spaces using body movements and gestures without wearing additional aides (e.g., data gloves and body markers) or handling remote controllers. Motion sensing technology has been around for a long time at prohibitive costs for the consumer market, and was consequently exploited only in niche fields. Today low cost devices and freely downloadable SDKs are available, so that TMI applications are becoming more popular and their development not requires much more that standard programming expertise. Most of the commercial TMI applications are for the entertainment market [14], but an increasingly vast amount of experimentations have taken place also in other domains.

Today the arena includes a variety of TMI applications: to support learning or motor rehabilitation (e.g., for disabled children [2]); to control of remote robots [5]; to interact with distant displays without the drawbacks of touch surfaces, e.g., when exploring contents in public spaces [7] or to access a patient's information in the sterile conditions of ORs [1].

In our research, we are interested in using TMI for *content intensive multichannel web applications,* i.e., applications that support exploration of *large* amounts of web-based multimedia information on different platforms. We have already witnessed an evolution in the delivery of web-based contents, from a single stationary device (desktop) to multi-devices (desktop + mobile), and this process has been widely studied [8]. Motion sensing technology can trigger the next step in this "going multi-channel" process, exploiting motion-sensing platforms as an additional channel for the delivery and fruition of web-based contents that takes advantage of the specifics of TMI.

Including TMI in the arena of multi-channel systems raises a number of challenges and represents a promising research field. Many issues are still open even for stand-alone TMI applications. Creating TMI systems as migration from pre-existing systems increases complexity, in terms of design and technology, while the current experience in this field is very scarce. Indeed, the few examples in the current state of the art exploit limited multichannel features. Compared with the desktop and mobile versions, the contents available on TMI interfaces are downsized, navigation topologies are oversimplified, software architectures are defined from scratch, and there is marginal sharing of information structures and implementation components with the related applications available on other devices. As a result, these applications do not fully capitalize on the benefits of a migratory approach, i.e., the provision of a coherent, integrated user experience across different platforms and the reduction, via data and software reuse, of development and maintenance costs

This paper focuses on the design issues related to the evolution from desktop to TMI interfaces. We propose a model based approach that helps developers maximize the reuse of content structures while exploiting a systematic mapping from content structures to interaction tasks to body movements and gestures. To exemplify our method, we present a case study in the e-culture domain.

2. THE IDM++ MODEL
Our model, called IDM++, is a revision of IDM – Interactive Dialogue Model, which has been developed in the web engineering community as a tools to represent design decisions for content-intensive web applications delivered on desktop or mobile platforms [3]. As in IDM, the IDM++ design process

AVI '14, May 27 - 30, 2014, Como, Italy

http://dx.doi.org/10.1145/2598153.2598194

comprises three main activities: Conceptual Design, Logical Design, and Presentation Design. While these activities are typically performed iteratively, each of them addresses different aspects of the application under design, at progressively lower levels of abstractions. Conceptual Design defines the general semantic features of the information delivered to the user regardless the actual characteristics of the delivery platform. It is the highest-level activity and is channel-independent. Logical and Presentation Design provides specifications at a progressively lower level of abstraction and is channel-dependent. In these phase, design decisions are taken on the basis of a variety of factors such as the constraints imposed by the type of display available on a given channel (e.g. screen size), the interaction mode (e.g., mouse, (multi) touch, touchless motion-based), and the typical situations of use (e.g., on the move, at home or working places, in public places). The IDM++ extensions to IDM mainly concern Presentation Design and are originated by the need of modelling the specifics of the interaction paradigms available in different devices, including TMI. The rest of this section provides a brief mention of the primitives for Conceptual and Logical Design (the reader is referred to [3] for details) and offers a more extensive discussion of IDM++ features for Presentation design.

During Conceptual Design, the designers defines the so called C-schema of the application, specifying what the application deals about in terms of the following primitives: i) Single Topics (Topics for short) and Types of Topics; ii) Relevant Relationships; iii) Single Groups of Topics (Groups for short) and Group Types. C-schemas have some similarity with Entity Relationship (ER) Schemas for databases but have a higher level of abstraction. In addition, they employ a number of constructs not available in ER: for grouping how subjects that are of interest for the user and for content structures that have single instances ("Group Types", "Groups", and "Topics").

Logical Design (resulting in the so called L-Schema) specifies the characteristics of the content associated to C-schema structures, splitting of Topics and Types of Topics into Components, and defining size, cardinality of such elements, media (e.g. audio, text, images, video, animation) of the Components, Groups, and Types of Groups.

Finally, Presentation Design crafts the actual interface and specifies how users interact with contents. The basic primitives of Presentation Design are Page, Page Type, Interaction Task, and User Action.

Pages and *Page Types* model "containers" of information in the visual interface and define which information structures in the L-schema are delivered "in one shot" on the device display.

Interaction Tasks define user's intentions for interacting, i.e., "*what*" the user wants to achieve while seeing a Page or an instance of a Page Type. IDM++ distinguishes between In-the-Small Interaction Tasks, In-the-Large Interaction Tasks, and Functional Interaction Tasks:

In-the-Small Interaction Tasks concerns the user's consumption of information *inside the current page* (i.e., revealing a hidden part of the contents, enlarging/shrinking an image or a map, playing/suspending a video or an animation).

In-the-Large Interaction Tasks capture the users' intention of exploring contents on a Page *different* from the current one, involve a Page transition, and are of different types.

Group Tasks concern transitions to Pages in the Group of the current Page (e.g., to the "next", "previous", or "root" Page in a group).

Jump Tasks allow the user to directly access the "home", or specific sections of the application.

Back Tasks support the return to the last visited Page.

Structural Tasks concern transitions to Pages containing contents of the same Topic or Type of Topics. *Relational Tasks* concern transitions to Pages of a different topic which is related via a semantic relationship to the one of the current Page.

Finally, *Functional Tasks* are less related to the semantics of information structures and concern launching and closing the application, or editing data entry fields or forms.

User Actions define the concrete actuation of Interaction Tasks, i.e., *"how"* the user performs the different interaction tasks and which instruments she uses (e.g., mouse, fingers, body). Examples are: "clicking" a button (with the mouse), "tapping" (with the finger) a control element, "dragging" (with mouse or finger), "swiping" (with fingers or arms).

A P-IDM schema is defined by the following elements:

- A set of Pages and Page Types
- A set of Interaction Tasks
- A set of User Actions
- A mapping from L-schema to the set of Pages and Page Type;
- A mapping from each Page or Page Type to the set of Interaction Tasks
- A mapping from the set of Interaction Tasks and the set of User Actions.

3. Discussion

In the design of TMI applications resulting as migrations of pre-existing applications, the starting point can be the IDM++ specifications of the desktop and mobile versions. While the C-schema is largely invariant across platforms, the L-schema and Interaction Tasks should be defined by adapting the L-schema and P-schema of the desktop version to the requirements and situations of use of the TMI version. Some Interaction Tasks of the existing versions can be hardly replicated because of technology limitations or contextual constraints. Editing Tasks, for example, might be removed. In principle, they could be made available by providing a virtual keyboard or hand-writing in the air, but both solutions have problems in precision and fatigue. Enabling voice commands could be an alternative, but not in crowded environments.

The most challenging design task is the definition of appropriate gestures (User Actions) to be mapped to each Interaction Task in the P-schema. For desktop applications, User Actions basically consists on *mouse-enabled selection & activation of control visual elements* (e.g., click on navigation links), hence the mapping "Interaction Tasks → User Actions" is trivial. In principle, in the migration to a touchless platform a similar simple mapping can be used, preserving all control elements of the desktop interface and replacing the mouse pointing mechanism with a *pointing gesture*. Pointing is a relatively intuitive and standardized gesture, but this solution does not take advantage of the full potential of the TMI paradigm. A more challenging approach is to adopt, at some degree, alternative body movements that convey the semantics of interaction tasks without the need of being coupled with control visual elements (which can be removed). Still, a touchless gestural language can hardly be defined to be the best one for any set of interaction tasks and any application. Human gestures and movements have ambiguous semantics, which may differ across individuals and cultures [6], and are seldom "natural" in a strict sense [9][10][11][12][13]. In particular, it is difficult to define non-pointing gesture for Semantic and Structural Tasks that have a domain-dependent meaning and are specific of each application. The User Actions associated to these Tasks would involve very special, ad hoc movements to be explained explicitly and to be learned from scratch. Gestures (different from pointing) are more appropriate for interaction tasks that are "domain independent". The meaning of tasks such as "revealing a hidden part of the contents", "zooming in/out", "next/previous/up" (Group Tasks), "home" (Jump Task), Undo, depends on the very nature of contents and navigation structures that users experience in all content intensive web applications and in all devices. In the migration from desktop to touch interfaces, most applications have removed the control buttons corresponding to these tasks and replaced mouse click with gestures (e.g., tap, drag, swipe, pinch, touch&hold). These gestures have progressively become more and more standardized by effect of the widespread use of devices and applications adopting them. A similar process may eventually take place also for TMI, with the same touchless boy actions for the above tasks meaning the same things in different systems for the majority of users.

In this overall scenario, it is worth noticing that the definition of the "best" movements is not the end of the design story. These decisions must be mediated with technological constraints. Designers must evaluate the feasibility of the interaction choices against the characteristics of the implementation platform and the technology available to recognize movements and gestures, revising some decisions if needed. In particular, the performances of algorithms are of primary importance: low performance in motion recognition and processing induce delayed feedbacks to users' actions, which can have dramatic effects on the quality of the user experience.

4. EXAMPLE

In order to exemplify some of the concepts of our model based approach, we shortly discuss the case study of the multichannel web application about the history and activities of our department ("DEI History – The history of the Department of Electronics and Information"). The application (http://www.storia.dei.polimi.it/) describes the past of our department and its activities through videos, textual narratives, and voice interviews (Figure 1).

Figure 1: Desktop Interface of DEI History Application.

The TMI version, to be installed in the hall of our building, provides the same contents and information architecture as the desktop version (Figure 2) but have a revised layout (Figure 2). Movements and gestures have been defined after a number of elicitation sessions overall involving 30 users. They were presented with the wall projection of the desktop application, which was remotely controlled by a researcher of our team, and were asked to try movements and gestures that could better express the different interaction tasks.

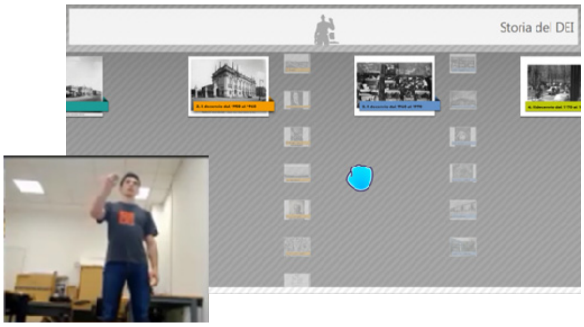

Figure 2: TMI Interface of DEI History Application.

The gestures are based on *Static* and Dynamic *Position* [4].

Static Gestures

- *45 Degrees Arm*: the user stretch his left arm up to 45 degrees and maintain this position for 2 seconds
- *Hands on Head*: the user raise both his hands up

Dynamic *Gestures*

- *Horizontal Swipe*: this gesture is composed by two sub-gestures: the Left-to-Right Horizontal Swipe (LtR) and the Right-to-Left Horizontal Swipe (RtL). Both of them consist of a horizontal movement of the user hand from a starting point far from the body to an end point close to the body. The LtR gesture can be performed only by the left hand, while the RtL only by the right one. The hand movement should not be too fast or too slow and it should not have considerable variation on the vertical axis.
- *Zoom*: this gesture is similarly composed by two sub-gestures: the zoom-in and the zoo-out. Both of them consist in the horizontal and symmetric movement of both hands. For the first type, hands are distant while for the second kind, hands get close. Hand movements should be monotonic and they should not have significant variation on the vertical axis.
- *Grab*: this gesture is recognized when the user close one of his hands; this gesture is independent from the hand position.
- *Push*: the user brings ahead and back one of his hands; also this gesture doesn't suffer from the position constraints.

In addition, we use user *position*, i.e., *proximity* to *the display,* to understand the user's intentions in front of the screen (Figure 3).

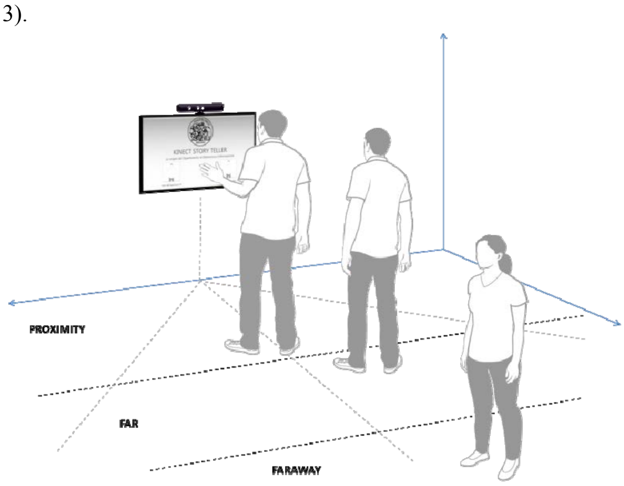

Figure 3: Proximity areas in front of the screen.

We divided the area in three different regions and we associated different system behaviors related to the interpolation of user position and gestures. The *Proximity area* is the closest zone in front of the screen. In this area user can interact with the system using static and dynamic gestures. The *Far area* is the intermediate zone. In this area user cannot interact using gestures because of the high distance from the screen, but she is attracted and encouraged to approach closely to the screen. Since the *Faraway area* is the most distant zone, the user is too far to be invited to come close to the screen.

Examples of the mappings between Interaction Tasks and User Actions in the desktop and TAI versions are shown in Table 1.

Table 1: Interaction Task → User Action Mapping

Interaction Task	Interaction Task Type	User Action: TAI Application	User Action: Desktop Application
Start Interaction	Functional	Hands on Head	Web address edit
Application Reset	Functional	User in "Faraway" area or further away	Web browser closure
Horizontal/ Vertical page scrolling	In the small	Grab + horizontal/ vertical hand movement	Mouse click on specific button
Overview/ Detailed visualization	In the small	Zoom-in / Zoom-out	Mouse click on specific zoom in/out button
Start/Stop videos	In the small	Push	Mouse click on specific button
Full screen visualization	In the small	Moving from "Proximity" to "Far" area	Mouse click on specific button
Normal screen visualization	In the small	Moving from "Far" to "Proximity" area	Mouse click on specific button
Next/Previous Chapter	In the Large	RtL / LtR Horizontal Swipe	Mouse click on specific button
Guided Tour	In the large	45 Degrees Arm	Mouse click on specific button

5. CONCLUSIONS

Our work provides a contribution to the current research on touchless motion based interfaces (TMIs) from a web engineering perspective. We focus on TMI systems that are regarded as components of a multichannel content intensive web application, i.e., as touchless versions of pre-existing systems available on desktop or mobile platforms. We provide a design model (IDM++) to allow designers to specify their solutions at a conceptual level while taking into account the characteristics of multichannel applications and the features of each specific interaction paradigm. IDM++ is intended to provide a conceptual framework that can help designers to master the complexity of the migratory process from desktop to TMI interfaces, to make it more systematic and better organized, and to improve communication between designers and developers. Our experience, gained in the development of a number of multichannel content intensive web applications in various domain (one of which reported in the paper as case study), has shown the effectiveness of IDM++ as a methodological support in the development process. Still, a number of intrinsically critical design issues remain open in the arena of TMI application design, mainly in relationship to the role of this interaction paradigm in modern web systems, the naturalness of touchless gestures and body movements, and the need of standardizing them.

6. Acknowledgments

This work is partially supported by the EC commission through EIT ICT Labs Project 14183/2014 "Street Smart".

7. REFERENCES

[1] Ali Bigdelou, Loren Arthur Schwarz, N. N. An adaptive solution for intra-operative gesture-based human-machine interaction. Proc. IUI 2012, ACM (2012), 75-84

[2] Bartoli L., Corradi C., Garzotto F., Valoriani M. 2013. Exploring Motion-based Touchless Games for Autistic Children's Learning. Proc. ACM IDC (Interaction Design and Children) 2013, 102-111

[3] Bolchini D, Garzotto F. 2007. Designing Multichannel Web Applications as "Dialogue Systems": the IDM Model. In: O. Pastor, G. Rossi, D. Schwabe, (eds.). Web Engineering: Modelling and Implementing Web Applications. Springer London, 193-219

[4] Baudel T. and Beaudouin-Lafon M. 1993 Charade: remote control of objects using free-hand gestures. CACM, 36(7):28-35, July 1993.

[5] Benavidez, P., and Jamshidi, M. 2011. Mobile robot navigation and target tracking system. Proc. SoSE 2011 , 299 –304.

[6] Cassell, J. 1998. A framework for gesture generation and interpretation. Computer Vision in Human-Machine Interaction (1998), 191–215.

[7] Kuikkaniemi, K., Jacucci G., Turpeinen M., Hoggan E., Muandller, J. 2011. From space to stage: How interactive screens will change urban life. Computer 44, 6 (2011), 40 – 47.

[8] Mori G., Paternò F., and Santoro C. 2004 Design and development of multidevice user interfaces through multiple logical descriptions. IEEE Trans. Softw. Eng., 30(8), August 2004, 507-520.

[9] Grandhi, S. A., Joue, G., Mittelberg, I. 2011. Understanding naturalness and intuitiveness in gesture production: insights for touchless gestural interfaces. Proc. CHI 2011, 821-824. ACM.

[10] Malizia A. and Bellucci A. 2012. The artificiality of natural user interfaces. CACM 55, 3 (March 2012), 36-38.

[11] Nielsen, M., Moeslund, T., Storring, M. Granum, E. 2004. A Procedure for developing intuitive and ergonomic gesture interfaces for HCI. Proc. Gesture-Based Communication in Human-Computer Interaction: 5th Int. Gesture Workshop, GW 2003, LNCS 2915, Springer, Berlin (2004), 409-420.

[12] Norman, D. A, Natural user interfaces are not natural. interactions, 17 (3), May + June 2010

[13] Norman, D. A, Nielsen, J. Gestural interfaces: a step backward in usability. interactions, 17 (5) September - October 2010

[14] Shan, C. Gesture control for consumer electronics. In Multimedia Interaction and Intelligent User Interfaces, L. Shao, C. Shan, J. Luo, and M. Etoh, (eds.), Advances in Pattern Recognition. Springer, 2010, pp. 107–128.