



Abstract - Graphical user interfaces have been a key factor in the digital revolution by making computing accessible to millions of users, but they were not designed to handle the complexity and diversity of information and contexts of use that we see today. My goal is to develop a new paradigm for interacting with computers based on my concept of **Instrumental Interaction**. Tools or instruments are a natural way to interact with the real world, and will serve as a powerful metaphor to interact with on-line information. An instrument reifies interaction: it turns an interaction into a meaningful object for users, designers and developers. For example, a pen reifies the act of writing. Instruments can be freely combined, substituted and even used with objects for which they were not designed. I envision a future where large, monolithic and closed applications are replaced by a rich ecology of instruments and information containers that can interoperate, giving users the power to shape their own environments. At the **empirical level**, I will study why and how instruments can be natural and efficient, based on the psychological theory of affordances. At the **conceptual level**, I will create a general framework for interaction in distributed environments featuring multiple devices, interactive surfaces and users. At the **engineering level**, I will develop the appropriate programming abstractions and software architecture to develop a full-scale prototype validating the key ideas of the project. The instrumental paradigm can radically improve today's graphical user interfaces. It simplifies interaction in complex environments while increasing the power and flexibility of interactive systems.

Introduction and motivation

The digital revolution has put computers in the hands of millions of people and equals in magnitude the industrial revolution or the invention of the printing press. One of the critical ingredients of this revolution was the advent of graphical user interfaces (GUIs) that allowed non-computer specialists to perform complex and diverse tasks. Building upon seminal work by Sutherland [40] and Englebart [16] in the sixties, GUIs were invented more than 30 years ago at Xerox PARC to replace command-line interfaces. They were first theorized and studied by Shneiderman [38], Hutchins, Hollan, Norman and Draper [22][33] and were later popularized in the Macintosh and Windows operating systems. GUIs are now by far the dominant paradigm for interacting with computers.

Despite this tremendous success, GUIs are showing clear signs of limitations as they face the ever-increasing reach of the digital revolution [36][30]. First, **GUIs do not scale well to large amounts of data**. For example, interacting with a search engine by entering a query in a single-line text field and viewing the results as a paginated list is like looking at the world through a pinhole. As the amount of information grows exponentially, be it email, the web, sensor-based data, images or documents, future user interfaces must address this scalability issue. In particular, the powerful solutions offered by Interactive information visualization [12] to visualize and analyze large amounts of data must be fully integrated with general interactive capabilities.

Second, **GUIs do not transfer well to new hardware platforms**. GUIs were optimized for specific input/output devices, namely the mouse, keyboard and screen of the personal computer. As more diverse platforms become widely available, including mobile devices today and large interactive surfaces such as tables and walls tomorrow, the basic building blocks of GUIs need to be reinvented. More importantly, users want to combine these various devices into integrated interactive environments where information and processing can fluidly move from one device to the next, where devices can control each other, and where groups of users can easily share data and processes with each other. The "ubiquitous computing" vision outlined by Mark Weiser two decades ago [43] has had a strong influence in the field of human-computer interaction, but so far GUIs have failed to fulfill it.

Finally, **GUIs are overly rigid**, especially for open-ended tasks where users define the problem at the same time as they solve it and want to create their own tools, adapted to their needs and skills. Inventions such as the spreadsheet have shown how built-in flexibility leads to unanticipated uses and innovation by the end-users themselves. Co-adaptation [28] of users and computers underlies many of the novel uses of computers, such as the recent advent of social networking systems, but is poorly supported by today's GUIs: In their current installment as closed, independent applications that barely communicate with each other except through limited drag-and-drop and copy-paste, GUIs are too monolithic to afford the flexibility, plasticity [14], viscosity [34] and reconfigurability [39] required to spur novel uses of computer technology.

The goal of this project is to replace GUIs with a new interaction paradigm that both builds on their strengths and challenges their fundamental assumptions. As with the original work that led to GUIs, fundamental research is needed to create and conceptualize this new paradigm, to assess its power, and to engineer its building blocks without the constraints of legacy systems.

The vision for a new paradigm

My vision for a new paradigm is to make interactions first-class objects from the perspective of the user, the designer and the developer. This is achieved by reifying the concept of interacting with an object into a *tool*, or *instrument*, that the user manipulates directly. From the user's perspective, a tool is a metaphor of a common way to manipulate objects in the real world. From the designer's perspective, designing an interface becomes designing a set of tools. From the perspective of the developer, programming an interactive application consists of creating tools and assembling them.

The key observation behind this vision is that tools mediate many of our interactions with the real world and that this can serve as a powerful paradigm for interacting with computers and on-line information. The power of tools, when they are well designed, lies in humans' ability to *internalize* them and consider them as natural extensions of their bodies. The tool then becomes "transparent": when writing with a pen, the pen is part of the user's hand and does not get in the way of writing – unless it runs out of ink. When operating a familiar device such as a car, a user can concentrate on the task at hand, i.e., driving, rather than on the particulars of operating specific levers, dials or buttons.

The goal of this project is ambitious: to demonstrate that an interaction paradigm that is based on a generalized notion of tool significantly outperforms current graphical user interfaces for interacting with large and complex data. Separating tools from the objects they operate on can result in greater flexibility and increased power. Tools can be combined and used in ways that were not anticipated by their designers. Generic tools can support standard tasks, while specific tools can be created for specific purposes. Designing a large ecology of tools can support different user populations and contexts of use, including environments with multiple interaction devices and multiple simultaneous users.

The history of mankind is punctuated by inventions for externalizing our physical and intellectual abilities. Writing and later the printing press greatly reduced the need to memorize and orally transmit information, while the industrial revolution created machines to manufacture objects out of materials that are difficult or impossible to manipulate by hand. The computer and the Internet are the most recent such major inventions, able to store, transmit and process more information than could ever fit into our heads. A search engine, for example, externalizes our ability to look for a document, saving time for more productive cognitive tasks.

All these inventions share a common trait that is also unique to humans: the creation and use of tools, instruments, machines and devices as intermediates between their users and these new technologies. While writing can be done with a finger in the sand, very early it was performed with a stick, and later with pens and pencils. Printing requires a press, a complex machine that is operated through a variety of

levers and dials. Similarly a search engine requires a dedicated tool to both specify the query and browse the results.

The notion of tool is not new to computer interfaces. Many existing applications leverage the tool metaphor. For example, many drawing and painting applications feature a tool palette where users can select brushes, pencils, colors, etc. However these tools are limited to a single application and are bundled with it: tools cannot be shared among applications, nor can they be added or replaced dynamically. A critical aspect of this project is to challenge the notion of application as the user-level building block of computer environments. Instead, the proposed approach will leverage the notion of *substrate*, to hold information, and *instrument*, to manipulate it. The resulting interaction model will enable a paradigm shift similar to that of moving from command-line interfaces to graphical user interfaces.

The next section reviews the state of the art in human-computer interaction from the perspective of interaction paradigms and supporting technologies. Then the concept of instrumental interaction is introduced. The following three sections describe the core of the project: the study of the instrumental paradigm at the empirical, conceptual and engineering levels. The last section analyzes the risks and potential benefits of the project.

State of the art of HCI research in interaction paradigms

Research on graphical user interfaces over the past 30 years has proceeded along four main strands: conceptual research on interaction models, empirical and theoretical research on human performance and cognition; creation of novel interaction techniques; and software models and tools for developing GUIs.

The conceptualization of the graphical user interfaces that started to appear at the end of the seventies was first conducted by Shneiderman [38], who coined the term “Direct Manipulation” to distinguish these interfaces from the then dominant command-line interfaces: rather than referring to objects and commands by names, GUIs allowed users to directly point at objects, such as icons, and commands, such as menu items, on the screen. Norman, Hollan, Hutchins and Draper, among others, studied the benefits and problems of direct manipulation interfaces in a seminal book [33]. However, as direct manipulation interfaces became quickly adopted, they became taken for granted and few articles were later published on the underlying interaction model of GUIs.

Research on empirical and theoretical models of human performance and cognition, on the other hand, thrived. Card conducted seminal work [11] on the performance of the mouse and other pointing devices, importing the experimental approach used in psychology to human-computer interaction. Fitts’ Law, in particular, proved extremely powerful to model and predict human performance for pointing tasks [29]. To this day, novel interaction techniques (see below) are systematically tested empirically to demonstrate their advantage over existing techniques. Together with Moran and Newell, Card also studied the cognitive aspects of human-computer interaction, introducing the GOMS family of models [13]. Other cognitive models, e.g. Barnard’s ICS [3], were also developed, with the goal of describing and predicting task performance in GUIs and other types of interfaces.

The above research in turn led to a long series of novel interaction techniques to improve the vocabulary of GUIs. This has been by far the most active part of recent research in GUIs. For example, research on improving the simple task of pointing at a target on the screen has generated a constant stream of innovations over the past 20 years (see Balakrishnan [2] for a review). Sadly, many such techniques, while proven to be far superior to existing ones, have not yet made it into mainstream interfaces. For example, marking menus [27], where the items are laid out radially, outperform standard linear menus by a factor of three or more, yet they are only used in a handful of commercial applications. A major reason is that, as explained above, research on the conceptual model of GUIs stopped early on, resulting in a lack of framework to integrate these novel techniques into existing interfaces.

Another key reason why novel techniques have not successfully made it out of the lab is the lack of proper software tools and programming abstractions. Research on User Interface Management Systems (UIMS) was very active in the early eighties [35]. In the nineties, software engineering approaches led to architectural models such as MVC [26] and PAC [14] and software tools such as user interface toolkits and generators (see [31] for a survey). These were quickly adopted by industry, e.g. NeXT's (and later Apple's) Interface Builder. Maybe this adoption was too fast, as many such tools lacked the flexibility required to easily add new interaction techniques, thus making it difficult for developers to integrate them into their interfaces.

At the risk of oversimplifying, research on GUIs over the past 20 years has concentrated on novel techniques at the expense of conceptualization, resulting in stereotyped applications that do not really follow the principles of direct manipulation but instead force users into cumbersome sequences of seemingly unrelated actions to perform conceptually simple tasks. Indeed, Norman's call for developing a conceptual model underlying any interactive application [32] is sorely ignored by many interface designers. However one should also recognize that research has failed to operationalize this and other findings into usable abstractions and tools for designers and developers.

In parallel with the above work on GUIs, research in human-computer interaction has produced a number of other interaction paradigms, triggered both by a desire to break out of desktop-based interaction and by new technology that made alternatives to GUIs possible. These alternative paradigms can be classified in two broad categories: "intelligent" interfaces and "tangible" interfaces.

Intelligent interfaces seek to delegate the details of the task that a user wants to carry out to the computer. By contrast with so-called "*first-person interfaces*" such as command-line interfaces and GUIs, where users have to carry out all the steps of a given task by hand, intelligent interfaces are "*second-person interfaces*" where the user delegates the task to the computer. This approach is strongly influenced by research in Artificial Intelligence and includes avatars and other agent-based interaction. Such interfaces are often based on a conversational mode of interaction where the user interacts with the computer using natural or semi-natural language. As such, intelligent interfaces build on humans' language skills more than on their sensory-motor skills.

Tangible interfaces, on the contrary, build on human sensory-motor skills and extend the notion of first-person interface by embodying interaction in the real, physical world. Weiser's vision of Ubicomp (ubiquitous computing) states that every surface could potentially be interactive, affording both display and input of information [43]. His group demonstrated this vision by building interactive tabs (i.e., badges), pads and whiteboards that could be used to share and cooperatively edit documents. Ubicomp has become a major driver for rethinking research in mobile and collaborative systems [1]. The original concept of Augmented Reality¹ as introduced by Wellner, Mackay and Gold [44] sought to embed information and computation into everyday objects, blurring the distinction between physical world and on-line world. It also has a deep impact on research in human-computer interaction, giving rise to a number of variants such as tangible interaction [19], where physical objects are used to interact with on-line information. More recently, Dourish theorized the notion of embodied interaction [15] and Jacob et al. introduced Reality-Based Interfaces [23]. Both approaches are grounded in an analysis of human skills and capabilities in relation with their environment.

Intelligent interfaces and tangible interfaces are not incompatible. Indeed, the vision for Ambient Intelligence that has been driving many recent calls for projects by the European Union gathers elements of both. A key difference however is that while intelligent interfaces are based on human-to-human modes of communication that are fairly well understood (or at least described), tangible interfaces afford novel forms of interaction that need to be created from scratch. This requires extensive empirical work to

¹ Augmented Reality was later restricted to systems that overlay virtual objects onto the real world using technology borrowed from virtual reality. The original notion of Augmented Reality is now best known as Mixed Reality.

create and test prototypes before being able to abstract out an interaction model. Indeed, apart from some rare exceptions such as Ullmer and Ishii's token+context model [42], the above-mentioned approaches have not in general been operationalized into interaction models,

In summary, while interaction paradigms have been a driving force behind most research in human-computer interaction, there has been relatively little research to identify and study them as such, particularly for first-person interfaces. Whether it is the direct manipulation paradigm that drives the vast majority of today's interfaces or the tangible interaction paradigm that underlies the most recent visions, research has mainly focused on new ideas and proofs-of-concept rather than concepts and abstractions and their operationalization into real systems. This lack of abstraction and conceptualization translates into poor tools for developing interfaces, which in turn slows down adoption by industry.

The instrumental paradigm

The goal of this project is to develop a first-person interaction paradigm based on the concept of tool, or *instrument*, that mediates interaction between the end-users and the on-line objects they want to manipulate. The fundamental idea is to build on humans' natural skills of using (and creating) tools to interact with physical objects in the real world.

Under today's GUI paradigm, interaction is carried out within the confines of *applications*. These include traditional, full-fledged desktop applications, but also the more limited "apps" that users can now install on mobile devices and the web applications accessible through a Web browser. An application embodies both a type of data that the user can manipulate and the interaction methods to carry out these manipulations. Applications act as independent *silos*, making it difficult or impossible to exchange either objects or interactions with other applications. This results in the duplication of functionality across applications, with inconsistent interfaces for similar functions. For example, while many applications feature a font or color selector, they are often different from one application to the next. Even when such interactions are standardized, their scope is limited to a single application: selecting the same color or the same font in two applications, e.g. a spreadsheet and a text editor, must be done explicitly.

Under the instrumental interaction paradigm, there are no applications. Data lives in a generalization of documents call *substrates*, and interaction is reified into *instruments* (or tools). Instrumental interfaces have the following properties:

- Any instrument can potentially be used with any document or part thereof, *even if it was not designed to do so*. For example, an annotation instrument should be usable with any document. If the document was not designed to support annotations, it may not be possible to attach them to a particular part of the document but only to the whole document.
- An instrument can easily be replaced by another instrument, *without the objects of interest even being aware of it*. For example, an instrument that displays data as a table can be replaced by one that visualizes it graphically, or an instrument for editing text with the keyboard can be replaced by one that uses voice recognition. Some instruments can be optimized to deal with large amounts of data, others for users with disabilities and others yet for highly trained specialists.
- An instrument can live on a different computer than that of the object it manipulates, therefore supporting a natural model of distributed interaction. For example, a color palette can be running on a mobile device so the user can select the color used to draw on a large whiteboard with a pen.
- Collections of instruments can be assembled freely and exchanged among users, providing an unprecedented level of flexibility in customizing the interface to the user needs. For example, graphic designers could create a bundle of highly specialized instruments for editing graphics or retouching photos while regular users would use a bundle of simplified instruments with more limited capabilities. However, all instruments would be usable with the same substrates.

This is not the first attempt to get rid of (or hide) the notion of application. The Xerox Star, which was the first commercial computer with a GUI, was based on the notion of document that could contain text, graphics, spreadsheets, plots, etc. However it was a closed system that was not designed to be extended

by third-party developers. In the late nineties, Apple developed OpenDoc, a document-based system running on the Macintosh. Documents were made of “parts” that could be text, graphics, etc. When clicking a part, the user would in fact switch to the application able to manipulate it. A similar facility, but with a less polished user experience, was available in Windows with OLE (Object Linking and Embedding).

The instrumental paradigm differs from these approaches by focusing on interaction rather than documents: unlike previous work, instruments turn interactions into first-class objects that can be stored, customized, combined, and shared. The goal is to demonstrate that this approach has more power than existing first-person interaction paradigms and yet can be easily adopted by users.

I described the principles of instrumental interaction in an article published in 2000 [5]. The article demonstrated the descriptive power of the model, i.e. how it encompassed a number of existing types of interaction, including graphical user interfaces and tangible interfaces. The article also described a number of properties of instruments that could be used to compare them. For example the degree of integration described how an instrument would match the degrees of freedom of the input device, e.g. a mouse, to the degrees of freedom of the operation applied to the object of interest, such as scrolling. Finally the article described how the notion of instrument could be used to generate new solutions to well-known problems, such as searching and replacing text. This article has been cited close to 250 times since it was published.

Since this original article, I have conducted several proofs of concept. The most important was the CPN/Tools application developed while visiting Univ. of Aarhus Denmark in 1998-2000. The application is a great success: it has been downloaded about 40,000 times since it was released (after I left Aarhus). It led to an article describing the design principles that we used to create instruments [6] and a description of the software architecture of the system [7]. While CPN/Tools was designed as an ad hoc application with a fixed set of instruments, my Ph.D. student Olivier Beaudoux explored how to better characterize and operationalize the principles of instrumental interaction and created an architecture model and toolkit embodying the concepts of instrumental interaction [4]. These were however limited to running instruments inside a single application.

In my most recent work I have studied instrumental interaction in a distributed environment in order to support Ubicomp environments and explore distributed multi-surface interaction [25][18]. I created the WILD platform that is used as a testbed for this (and other) projects. WILD features a 18’x6’, 131 million pixel ultra-high resolution wall display, a VICON motion-tracking system, a multitouch table and a number of mobile devices, including iPodTouch and iPads. The Substance framework that we have developed [18] is used to create multi-surface collaborative instrumental interfaces for use by scientists, including astrophysicists, biologists and neuroanatomists, who need to analyze large amounts of data.

Overall, this work has shown the validity of the concept, but in limited settings. The separation between instruments and manipulated objects and the absence of applications address the problems of GUIs outlined in the introduction: scalability can be achieved by designing instruments that support large data sets and information visualization techniques; platform diversity is supported by the ability to use instruments that take advantage of the platform’s interaction resources and by a natively distributed architecture; flexibility is supported by the independence between instruments and the objects that they operate on.

The next stage however involves a much more concerted effort to create a full-scale prototype and demonstrate not only the validity but also the power of the approach. The test-bed for such a full-scale prototype will be the WILD room and the DIGISCOPE project that I submitted to the French National Research Agency recent call for “Equipments of Excellence”. The goal of DIGISCOPE is to create a network of ten high-end interactive rooms, ranging from immersive virtual reality rooms to 3D power walls and multi-surface environments such as WILD. These rooms will be interconnected with high-end

telepresence audio-video systems. Targeted applications include scientific discovery, product life-cycle management, decision support and business intelligence, and education and training. With 10 academic partners from the wider Paris area and strong support from key industry partners, e.g. in the automotive, aerospace and energy areas, DIGISCOPE will provide a real full-scale stage to experiment with the instrumental paradigm².

The following sections describe the three main tasks of the project: validating the underlying empirical theory of instruments; developing the conceptual model of instrumental interaction; designing and creating an instrumental interaction toolkit to facilitate the development of instrumental interfaces.

An empirical theory of instruments

Gibson's ecological theory [17] introduced the notion of *affordance*, i.e. the perceived properties of an object for action: depending on its height and weight, a chair affords sitting as well as climbing or blocking a door. Affordances are properties of neither the object nor the subject (animal or human), but of the *relationship* between them. The ecological perspective on tools (or instruments) is that as long as the instrument is detached from the human (or animal) user, it behaves like any other object. However, once attached, i.e., by taking it in one's hand, the tool becomes part of the user's body and *changes the affordances of the environment*. A sheet of paper whose affordances include being used as a wedge under a wobbly table suddenly acquires the affordance of starting a fire when the user holds a match in the hand, or being written on when the user holds a pen. Conversely, the affordance of the sheet of paper for writing is conditioned by the affordance of an object to deposit ink or lead. The user may then seek this affordance in the environment and discover a pen, or a charred piece of wood. This perspective on the ecological approach to instruments illustrates the theoretical basis for the ability of instruments to be used in unanticipated ways. It will be complemented by other theories such as Distributed Cognition [20], Activity Theory [10] and Situated Action [39], which provide a rich framework for understanding human cognition.

The fact that an instrument is internalized as part of one's body is evidenced, in the context of graphical interfaces, by the success of pointing techniques that "cheat" with the real world. For example, the so-called mouse acceleration uses a non-linear mapping between the movements of the mouse and the corresponding movements of the cursor [24]. Yet few users actually notice this non-linear gain. The same is true when the gain is varied dynamically as a function of the position of the cursor: in our Semantic Pointing technique [8], the cursor gain varies as a function of the distance to the closest target, so that for a constant speed of the mouse, the cursor travels faster in "empty space" than near and on targets. The net result is that pointing performance is improved, but users *do not* notice the manipulation, as if they completely internalized the distorted geometry of the space created by the technique.

The goal of this task will be to provide empirical evidence of the theoretical hypotheses underlying the instrumental paradigm. Using a multidisciplinary approach, I will conduct controlled experiments to characterize the phenomenon of internalization and I will operationalize properties of instrumental interaction to compare and contrast instruments. This knowledge will provide a sound basis for developing the conceptual model of instrumental interaction.

A conceptual model of instrumental interaction

The conceptual model of interaction developed in my earlier work and outlined above is still far from being complete and sufficiently refined. In particular, it is still difficult to demonstrate the completeness and the power of the approach. Completeness is defined as the ability to do everything that existing (GUI) interfaces can do, although differently: Can instrumental interaction really cover the wide range of GUI applications, from document editing to mediated communication, from controlling physical devices to modeling and simulation? Can we establish the limits, if any, of the paradigm? Power, on the other hand, is defined as the ability to do things that current interfaces cannot do: Can instruments really be

² If DIGISCOPE is not funded, a scaled-down version will nevertheless be established with the partners' existing equipment.

independent of the manipulated objects? Can instruments really be used in unanticipated but interesting ways without breaking the objects? Only a full-scale prototype can provide convincing evidence, and such a prototype must build on a solid conceptual model.

The basic components of the model are *substrates* and *instruments*. Substrates are generalizations of documents that describe the data to be interacted with. While documents are static, i.e. they contain their own state, substrates can also contain intensional or procedural data, e.g. the records extracted from a database, the messages exchanged in a communication, the result of a search query or of a simulation that is run every time the document is accessed. How exactly can substrates support instruments is still an open question and one of the keys to the completeness and power of the paradigm. For example, instruments will need a unified way to access and keep references to the data stored in the substrate, even when it is described by intension or procedurally; Instruments may need to store extra state within the substrate, e.g. the positions of the icons in a file system; Multiple instruments operating on the same data will need to coordinate so as to avoid conflicts; etc.

Instruments reify interactions, i.e. they embed which user actions are valid and what their effect on the target object is. The ability for an instrument to operate on a piece of data is described by *protocols*: if the instrument and the information have compatible protocols, they can interoperate. A *formal protocol* specifies an interaction that was anticipated. For example, a formal protocol for changing the color of an object could be that the object has a “SetColor(c)” method. *Informal protocols*, on the other hand, specify situations where an interaction was not anticipated but is still possible. For example, a color tool may be able to act on an object that happens to have a property of type “Color”, or a method that takes a Color as parameter. Obviously, the object can offer no guarantee that setting the color property or calling the method will have the expected effect. But the system can let the user try and tag the given protocol as satisfactory or not for the given substrate, thereby enabling or disabling future attempts at using this instrument with objects of this type. Specifying exactly how protocols work is the second critical aspect of this task, as it will be the key to providing the richness of affordances that we enjoy in the real world.

A third component of the conceptual model is to envision how instruments are made available to the user, especially since there will be many different instruments available at any one time. Part of this consists of letting the user bundle instruments together and tying them to specific substrates. Here too, a full-scale prototype will be critical to demonstrate the validity of the approach.

An instrumental interaction toolkit

Assuming that the two tasks above provide the expected results, the third task will be to develop a set of software tools to create instrumental interfaces and a functional prototype. This is an ambitious goal because of the wide scope of the instrumental paradigm: it must support distributed, multiuser interaction on a variety of devices. Since it is not trying to support legacy software, everything must be recreated from scratch. In addition, programming interactions is notoriously difficult, as existing programming languages typically do not feature the proper abstractions, e.g. for event-driven programming.

The goal of the instrumental interaction toolkit is two-fold. One goal is to facilitate the dissemination of the instrumental paradigm, with the hope that third-party researchers and developers will contribute substrates and instruments. Such adoption would be the ultimate measure of the success of the project. The other goal is at least as important and consists of internal validation of the project: developing a toolkit requires the operationalisation of the key concepts of the conceptual model into software components. In that sense, a toolkit with good properties (“simple things are simple, complex things are possible”) is an empirical validation of the conceptual model it implements.

The implementation of the toolkit raises a number of architectural questions, such as: is each instrument and each substrate implemented as a separate process, making it easy to support distribution across multiple machines but raising the issue of efficient inter-process communication? Or do they all run in a single process for each machine, in which case distribution must be handled as a special case? How to

manage graphical rendering and input handling in a distributed fashion? To what extent do we build on existing toolkits and user interface frameworks? My earlier work in the INDIGO project [9] and the VIGO [25] and Substance [18] frameworks will provide precious insights, as well as previous work such as SubArctic [21] or XWeb [37].

In summary, this task will provide the ultimate proof of validity and power of the instrumental paradigm. Ideally it will be conducted in the context of the DIGISCOPE project, taking advantage of the unique infrastructure it will provide for local and remote collaborative work involving a rich ecology of interaction resources and a wide range of application areas.

Risk/benefit analysis and conclusion

This project is ambitious: few attempts have been made to create and test novel interaction paradigms in full scale, and fewer have made it beyond the laboratory. However, if successful, it may mark the beginning of a new era for interactive computation. I envision a future where large, monolithic and closed applications are replaced by a rich ecology of substrates and instruments that can interoperate, giving users the power to shape their own environments. This vision requires a new business model for software: While vendors would still sell bundles of instruments and substrates that replicate current desktop applications, users could mix and match them with those of other vendors. There would be a market for individual instruments and substrates: some simple and cheap, others more sophisticated or specialized for particular professions. The success of Apple's AppStore indicates that such a business could thrive.

Like any ambitious project, the project is risky. Some of the hypotheses underlying the instrumental paradigm may prove incorrect. The claimed flexibility, interoperability, openness to unanticipated uses may not scale or generalize to a useful subset of interesting situations. I am, however, optimistic, given the success of my earlier proofs-of-concept, that this project will instead lead to a deeper understanding of those limits and the potential scope of instrumental interaction. Another risk stems from the sheer quantity and variety of work to be performed. Not only do I have extensive experience in the empirical, conceptual and engineering aspects of interactive systems, but I can also greatly benefit from the world-class InSitu lab where the work will be conducted. The recent four-year evaluation of InSitu concluded that it is one of the rare groups in the world that have the skill set and vision to create such a breakthrough.

Finally, the project is timely. The predicted (but as yet unobserved) demise of the GUI and the desktop PC has focused attention on new platforms such as smartphones without considering the need for unifying principles of interaction. Today's landscape of user interfaces is becoming increasingly fragmented and inconsistent: Desktop applications compete with web applications and mobile apps, each living in its own closed world and requiring users to spend considerable time and energy synchronizing, transferring and consolidating data across devices and into the various clouds. Platforms are deeply incompatible, not only at the hardware and operating system level, but also in terms of the user's experience of even the most basic tasks. Individual users now routinely juggle multiple mobile, portable and desktop computers, increasing the need for a unified approach.

The instrumental paradigm offers supports such a unified approach, with simple and powerful interactions, a more consistent and productive user experience, and a more open environment for designers and developers. This project will conduct the empirical studies necessary to establish the theoretical foundations of instrumental interaction and then refine the conceptual model by developing and testing a new set of tools and a full-scale prototype that demonstrates the power and validity of this paradigm-changing approach to interactive computation. If successful, the instrumental paradigm will radically improve over today's graphical user interfaces, replacing them with a rich ecology of information substrates and interaction instruments.

References

- [1] Abowd, G.D., Mynatt, E.D., Rodden, T. (2002). The human experience [of ubiquitous computing]. *Pervasive Computing*, IEEE , 1(1):48-57.
- [2] Balakrishnan, R. (2004). "Beating" Fitts' law: Virtual enhancements for pointing facilitation. *International Journal of Human-Computer Studies*, 61(6):857-874.
- [3] Barnard, P.J. (1985). Interacting Cognitive Subsystems: A psycholinguistic approach to short term memory. In A. Ellis (Ed.), *Progress in the Psychology of Language*, Vol. 2, Chapter 6. London: Lawrence Erlbaum Associates, 197-258.
- [4] Beaudoux, O. & Beaudouin-Lafon, M. (2001). DPI: A Conceptual Model Based on Documents and Interaction Instruments. In *People and Computers XV - Interaction without frontiers* (Joint proceedings of HCI 2001 and IHM 2001, Lille, France), pp 247-263, Springer Verlag.
- [5] Beaudouin-Lafon, M. (2000). Instrumental Interaction: an Interaction Model for Designing Post-WIMP User Interfaces. *Proc. ACM Human Factors in Computing Systems*, CHI 2000, The Hague (The Netherlands), CHI Letters 2(1):446-453, ACM Press.
- [6] Beaudouin-Lafon, M. & Mackay, W. (2000). Reification, Polymorphism and Reuse: Three Principles for Designing Visual Interfaces. *Proc. Advanced Visual Interfaces*, AVI 2000, Palermo (Italie), ACM Press, pp 102-109.
- [7] M. Beaudouin-Lafon & H.M. Lassen (2000). The Architecture and Implementation of CPN2000, a Post-WIMP Graphical Application. *Proc. ACM Symposium on User Interface Software and Technology*, UIST 2000, San Diego (USA), CHI Letters 2(2):181-190, ACM Press.
- [8] Blanch, R., Guiard, Y., Beaudouin-Lafon, M. (2004). Semantic Pointing: Improving Target Acquisition with Control Display Ratio Adaptation", *Proc. ACM Conference on Human Factors in Computing Systems*, CHI 2004, Vienna (Austria), CHI Letters 6(1), ACM Press, pp 519-526.
- [9] Blanch, R., Beaudouin-Lafon, M., Conversy, S., Jestin, Y., Baudel, T., Zhao, Y-P (2006). Concevoir des applications graphiques interactives distribuées avec INDIGO. *Revue d'Interaction Homme-Machine (RIHM)*, 7(2):113-140.
- [10] Bødker, S. (1990). *Through the Interface: A Human Activity Approach to User Interface Design*. L. Erlbaum Assoc. Inc., Hillsdale, NJ, USA.
- [11] Card, S.K., English, W.K., Burr, B.J. (1978). Evaluation of mouse, rate controlled isometric joystick, step keys and text keys for text selection on a CRT. *Ergonomics*, vol. 21, p. 601-613.
- [12] Card, S.K., Mackinlay, J.D., Shneiderman, B. (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers.
- [13] Card, S.K., Moran, T.P., Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates.
- [14] Coutaz, J. (1987). PAC, An Object-Oriented Model for Dialog Design. *Proc. Human-Computer Interaction*, INTERACT'87, p. 431-436, Elsevier/IFIP.
- [15] Dourish, P. (2001). *Where The Action Is: The Foundations of Embodied Interaction*. MIT Press.
- [16] Engelbart, D.C. & English, W.K. (1968). A research center for augmenting human intellect. *Proc. Fall Joint Computer Conference*, AFIPS '68, p. 395-410, ACM Press.
- [17] Gibson, J.J. (1979). *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin.
- [18] Gjerlufsen, T., Klokmose, C., Eagan, J., Pillias, C., Beaudouin-Lafon, M. (2011). Shared Substance : Developing Flexible Multisurface Applications. *Proc. ACM Conference on Human Factors in Computing Systems*, CHI '11, ACM Press, in press.
- [19] Ishii, H. & Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. *Proc. Human Factors in Computing Systems*, CHI '97, p. 234-241, ACM Press.
- [20] Hollan, J., Hutchins, E., Kirsh, D. (2000). Distributed cognition: toward a new foundation for human-computer interaction research. *ACM Trans. Computer-Human Interaction*. ACM Press, 7(2):174-196.
- [21] Hudson, S., Mankoff, J., Smith, I. (2005). Extensible Input Handling in the subArctic Toolkit", *Proc. Human Factors in Computing Systems*, CHI '05, p. 381-390, ACM Press.

- [22] Hutchins, E.L., Hollan, J.D., Norman, D.A. (1985). Direct manipulation Interfaces. In *User centered system design*, D. A. Norman and S. W. Draper (Eds.), Lawrence Erlbaum Associates Inc, pp. 87-124.
- [23] Jacob, R., Girouard, A., Hirshfield, L.M., Horn, M.S., Shaer, O., Solovey, E.T., Zigelbaum, J. (2008). Reality-Based Interaction: A Framework for Post-WIMP Interfaces. *Proc. Human Factors in Computing Systems Conference, CHI '08*, pp. 201-210, ACM Press.
- [24] Jellinek, H.D. & Card, S.K. (1990). Powermice and user performance. *Proc. Human factors in computing systems, CHI '90*. ACM Press, p. 213-220.
- [25] Klokmose, C., & Beaudouin-Lafon, M. (2009). VIGO: Instrumental Interaction in Multi-Surface Environments. *Proc. ACM Conference on Human Factors in Computing Systems (CHI '09)*, ACM Press, pp 869-878.
- [26] Krasner, G.E. & Pope, S.T. (1988). A cookbook for using the model-view controller user interface paradigm in Smalltalk-80. *Journal of Object Oriented Programming*, 1(3):26-49.
- [27] Kurtenbach, G. & Buxton, W. (1994). User learning and performance with marking menus. *Proc. Human factors in computing systems, CHI '94*, p. 258-264, ACM Press.
- [28] Mackay, W.E. (1990). *Users and Customizable Software: A Co-Adaptive Phenomenon*, Ph.D. Thesis. Massachusetts Institute of Technology.
- [29] MacKenzie, I. S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7:91-139.
- [30] Moran, T.P. & Zhai, S. (2007). Beyond the desktop metaphor in seven dimensions. *Beyond the Desktop Metaphor: Designing Integrated Digital Work Environments*, Victor Kaptelinin & Mary Czerwinski (Eds), MIT Press, p 335-354.
- [31] Myers, B., Hudson, S., Pausch, R. (2000). Past, Present and Future of User Interface Software Tools. *ACM Transactions on Computer Human Interaction*, 7(1):3-28, March 2000.
- [32] Norman, D.A. (1988). *The Psychology of Everyday Things*. Basic Books.
- [33] Norman, D.A. & Draper, S.W, Eds. (1985). *User centered system design*, Lawrence Erlbaum Associates.
- [34] Olsen, D. (2008). Interactive viscosity. *Proc ACM symposium on User Interface Software and Technology, UIST '08*, ACM, p. 1-2.
- [35] Olsen, D. R., Buxton, W., Ehrich, R., Kasik, D., Rhyne, J., Sibert, J. (1984). A Context for User Interface Management. *IEEE Computer Graphics and Applications* 4(12):402-420.
- [36] Olsen, D.R. (1999). Interacting in chaos. *Interactions*, ACM Press, 6(5):42-54.
- [37] Olsen, D.R., Jefferies, S., Nielsen, T., Moyes, W., Fredrickson, P. (2000). Cross-modal interaction using XWeb. *Proc. ACM symposium on User Interface Software and Technology, UIST '00*. ACM Press, p.191-200.
- [38] Shneiderman, B. (1983). Direct manipulation: a step beyond programming languages. *IEEE Computer*, 16(8):57-69.
- [39] Suchman, L. (2006). *Human-machine reconfigurations: plans and situated actions*. Cambridge University Press.
- [40] Sutherland, I.E. (1963). SketchPad, a man-machine graphical communication system. *Proc. AFIPS SJCC*, Vol. 23, AFIPS Press, p. 329-346.
- [41] Thevenin, D. & Coutaz, J. (1999). Plasticity of User Interfaces: Framework and Research Agenda. *Proc. Human-Computer Interaction, Interact'99* (Edinburgh), p. 110-117, IFIP IOS Press.
- [42] Ullmer, B., Ishii, H., Jacob, R. (2005). Token+Constraint Systems for Tangible Interaction with Digital Information. *ACM Transactions on Human-Computer Interaction*, 12(1):81-118, ACM Press.
- [43] Weiser, M. (1991). The Computer for the 21st Century. *Scientific American*, 265(3):94-101.
- [44] Wellner, P., Mackay, W., Gold, R. (1993). Back to the real world. Special issue on computer-augmented environments. *Communications of the ACM*, 36(7):24-26, ACM Press.