# Supporting Interaction and Co-evolution of Users and Systems

Maria Francesca Costabile, Antonio Piccinno
Dipartimento di Informatica
Università di Bari
Bari, Italy
+39 080 544 3300
{costabile, piccinno}@di.uniba.it

Daniela Fogli
Dipartimento di Elettronica per l'Automazione, Università di Brescia,
Brescia, Italy
+39 030 371 5455
fogli@ing.unibs.it

Andrea Marcante
CNR - ITC
Tecnologie Informatiche Multimediali
Milano, Italy
+39 02 503 16330
marcante@dico.unimi.it

## ABSTRACT

Interactive systems supporting people activities, even those designed for a specific application domain, should be very flexible, i.e., they should be easily adaptable to specific needs of the user communities. They should even allow users to personalize the system to better fit with their evolving needs. This paper presents an original model of the interaction and co-evolution processes occurring between humans and interactive systems and discusses an approach to design systems that supports such processes. The approach is based on the "artisan's workshop" metaphor and foresees the participatory design of an interactive system as a network of workshops customized to different user communities and connected one another by communication paths. Such paths allow end users and members of the design team to trigger and actuate the co-evolution. The feasibility of the methodology is illustrated through a case study in the medical domain.

## Categories and Subject Descriptors

H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces – *Asynchronous interaction, Organizational design, Theory and models*

## General Terms

Design, Experimentation, Human Factors.

## Keywords

Interaction model, co-evolution, participatory design, usability.

## 1. INTRODUCTION

In the last years we have been working in the design of visual interactive systems supporting collaborative human activities. By studying people and the activities they perform in their daily work, we realized that in various domains, activities are performed by experts who do not constitute a uniform population, but they belong to different communities characterized by different cultures, goals, and tasks. In system design we must consider that user diversity arises due to: a) different culture, skill, specific abilities (physical and/or cognitive), tasks; b) different roles assumed by the user in performing work activities; c) different context of activity and geographical dispersion (of the community). For example, in the medical domain, neurologists cooperate with neuro-radiologists to interpret a Magnetic Resonance Image (MRI) and come out with a diagnosis; they are members of two different communities who must analyse and manage the same data set with different tools, on the basis of different knowledge they possess and from different points of view. In this activity, as in many others, members of different communities reach a common understanding and co-operate to achieve a common purpose [12].

As a consequence, interactive systems supporting people activities, even those designed for a specific application domain, should be very flexible, i.e., they should be easily adapted to specific needs of the user communities. Moreover they should even allow users to personalize the system to better fit with their evolving needs.

Many authors have pointed out an important phenomenon that must be considered in Human-Computer Interaction (HCI): the *user evolution*. Nielsen says in [15] that "using the system changes the users, and as they change they will use the system in new ways". More recently, Norman says in [17] that "the individual is a moving target". This means that a design of an interactive system may be good today, but no longer appropriate tomorrow. Once people gain proficiency in system usage, they would like to use the system in different ways and need different interfaces than those they required when they were novice users. These new uses of the system force the designers to evolve the system to meet these new needs. Therefore, it is more appropriate to speak about *co-evolution of users and systems* [2][5][8][11].

The work presented in this paper discusses a model of the Interaction and Co-Evolution processes (ICE model) occurring between human and system, which helps to identify the causes of usability difficulties affecting interactive systems. Even though this model is presented here for the first time, it was implicitly used to develop a design methodology to create flexible and tailorable interactive systems. User studies in real cases in the medical domain [11], the mechanical domain [9] and the geological domain [7], have highlighted the needs of end users and their difficulties with current technologies. From this, we have derived the main ideas of our design methodology and a better understanding of the interaction and co-evolution processes that are formalized in the proposed model.

Besides the emphasis on the ICE model, this paper discusses how the design methodology previously proposed in [9][11] leads to the development of interactive systems that allow end users to manage the co-evolution process.

The paper has the following organization. Section 2 discusses the sources of co-evolution and some examples from well known software environments. Section 3 introduces an original model of the interaction and co-evolution processes. Section 4 describes the design methodology, based on the model, to support the two processes. Section 5 presents the application of the methodology to a case study in the medical domain. Section 6 discusses some related works and concludes the paper.

## 2. CO-EVOLUTION OF USERS AND SYSTEMS

Co-evolution is an important phenomenon that designers cannot neglect anymore. Co-evolution stems from two main sources: a) users creativity, i.e., the users may devise novel ways to exploit the system in order to satisfy some needs not considered in the specification and design phase; and b) user acquired habits, i.e., users may insist in following some interaction strategy to which they become accustomed; this strategy must be facilitated with respect to the initial design.

An example of the first type is the integration of non-numerical data in spreadsheets, which was included in later versions of spreadsheets, after the observation that users frequently forced the spreadsheet to manage non-numerical data for data archiving and other tasks [16]. Other examples derive from observing how users learn to interact with web documents [2][3].

An example of co-evolution stemming from user acquired habits is offered by the strategy for saving in a new directory a file being edited. In earlier versions of many applications (e.g. those of the Microsoft® Office suite) after selecting the "Save as" command the user could create a new directory, which however does not become the current directory. Users are required a third command - open the new directory - before saving their file. In this editing situation, forcing the user to open the newly created directory is obviously inconvenient. Having recognized this contextual nuisance, more recent versions of Microsoft® Office applications co-evolved to encompass this user behaviour: when a new directory is created in the "Save as" context, it automatically becomes the current one.

In our work with end users, we found that several usability problems depend very much on the rigidity of the interactive systems. Users want systems able to take care of the changes occurring in their activities and/or in their organizational context. This leads to deeply re-examine the way interactive systems are designed by finding new theoretical foundations on which new approaches could be based. The following section presents the theoretical model on which our design methodology is based.

## 3. A MODEL OF INTERACTION AND CO-EVOLUTION PROCESSES

Model-based approaches to HCI attempt to identify and frame characteristics of the interaction process into a model. The model is used as a unifying framework to identify the causes of interaction difficulties affecting software systems, i.e. usability issues.

The seminal interaction model proposed by Hutchins, Hollan and Norman in [14] identifies the existence of semantic and articulatory distances in evaluation and execution as the primary sources of usability difficulties. The interaction process is determined by the human, that is a cognitive system, and a computer, that is a computing system. This model focuses on the human side of the interaction process.

Another interaction model, proposed by Abowd and Beale, highlights the problems arising on the computer side, i.e., capturing and interpreting the human actions [1]. In fact, to properly model the interaction process, we must also model the computing system.

This stance is also adopted by a model of HCI proposed in [4]. HCI is modeled as a cyclic process, in which the user and the interactive system communicate by materializing and interpreting a sequence of messages (the images on the screen in visual interaction) at successive points in time - $i(t_0)$, $i(t_1)$, ..., $i(t_n)$ -. These messages are subject to two interpretations: one performed by the user, depending on her/his role in the task, as well as on her/his culture, experience, and skills, and the second one internal to the system, associating the image with a computational meaning, as determined by the programs implemented in the system (see Figure 1).

The emphasis given in this model to these two interpretations explains a problem that arises in HCI, that is the *communication gap* between users and designers. The interpretation performed by the system reflects the designers' understanding of the task at hand, implemented in the programs that control the machine. Designers develop the interactive system and focus primarily on the computational and management aspects, rather than on the users' problems, the interaction language often being too general and machine oriented rather than situation and user oriented. Users need to perform their tasks by reasoning in accordance to their mental models, and to express this reasoning in notations

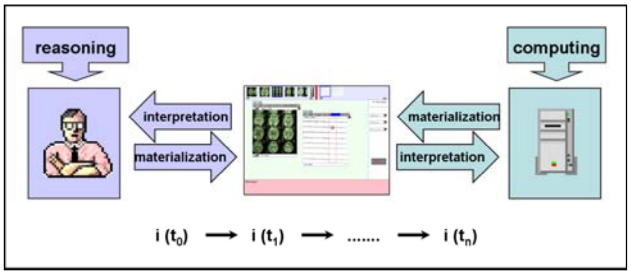


**Figure 1. The Human-Computer Interaction model.**

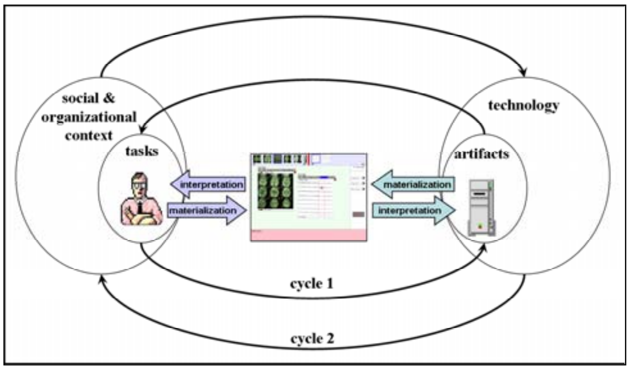


**Figure 2. The Interaction and Co-Evolution model.**

familiar to them. Unfortunately, traditional design approaches force users to communicate by means of computer-oriented notations, which are alien to their culture, generally not amenable to their reasoning, and often misleading for them. In this way, users are forced to break the continuity of their reasoning in order to translate and express their problems and solutions in the computerized language.

As we said in the introduction, in our experiments with end users, we found that several usability problems depended very much on the rigidity of the interactive systems, which are not able to take care of the changes occurring in users' activities and/or in their organizational context. A model-based approach to the design of usable interactive systems must then consider the co-evolution process as well. Indeed, in a working environment augmented by the support of software systems, two processes occur. The first process - the interactive use of the system to perform activities in the application domain - occurs in a short time scale: every activity is the result of a sequence of interaction cycles in which the user applies her/his intuitive knowing and reflects on the obtained results, gaining new experience. The second process is the co-evolution of user and system, which occurs during the use of an interactive system in a long time period.

We propose in this paper the ICE (Interaction and Co-Evolution) model that encompasses both interaction and co-evolution processes. The interaction process is modeled as in Figure 1, while the co-evolution process is modeled by refining the models in [8] and in [5]. Specifically, the task-artifact cycle in [8] implicitly referred to the co-evolution process by describing that software artifacts are produced to support some user tasks. However, such artifacts suggest new possible tasks so that, to support these new tasks, new artifacts must be created. This task-artifact cycle is denoted as "cycle 1" in Figure 2, indicating a first co-evolution cycle. Technology advances give computer scientists new possibilities of improving interactive systems once they are already in use: this leads to new interaction possibilities that might change end users working habits. For example, recently improved voice technology allows software engineers to add voice commands to their interactive systems and this might provide end users with a more easy and natural way to use the system. On the other hand, the user social and organizational context is evolving during time, requiring new tasks and/or different ways of performing tasks. Therefore, technology and social and organizational contexts repeatedly affect each other: this is represented in our ICE model with a second co-evolution cycle, denoted as "cycle 2" in Figure 2.

Software engineers are required to produce the tools to support the interaction and co-evolution processes. In other words, they must not only produce interactive systems supporting user activities, but also the tools that permit to evolve the system according to user evolution.

## 4. INTERACTIVE ENVIRONMENTS SUPPORTING CO-EVOLUTION

Our design methodology is extensively illustrated in [9][11]. It is briefly recalled here with the main intent to focus on its capability of supporting the co-evolution process.

In such a methodology, software environments are designed in analogy with artisan workshops. Artisan workshops are small establishments where artisans, such as blacksmiths and joiners, manipulate raw materials in order to manufacture their artifacts. At each step of their activity, traditional artisans can extract from a repository the tools necessary for the current activity and set back those ones no more needed. In this way, every artisan adapts the environment to her/his needs and has available all and only the tools needed in the specific situation. In analogy, a software environment is designed as a *virtual workshop*, in which the end user finds a set of (virtual) tools whose shape, behavior and management are familiar to her/him. Such an environment allows end users to carry out their activities and adapt environment and tools without the burden of using a traditional programming language, but using high level visual languages tailored to their needs. Moreover, end users get the feeling of simply manipulating the objects of interest in a way similar to what they might do in the real world. Indeed, they are creating programs, through which they later perform the necessary computations, without writing any textual program code. Obviously, while traditional artisans shape real supplies, end users shape software artifacts. For this reason we call these environments *Software Shaping Workshops* (SSWs) [9][11].

The SSW approach provides each end user sub-community with a workshop, called *application workshop*, which supports them in their daily work. An application workshop is customized to users' culture, background and skills, and can possibly be tailored by the users themselves, also by creating new artifacts [11]. Application workshops are not directly created (and successively evolved) by software engineers, but their design, development and modification are carried out, with a participatory approach [18], by an interdisciplinary team that, besides software engineers, includes representatives of end users and HCI experts. Each member of the team of experts uses a type of workshop, called *system workshop,* customized to her/his culture, background and skills, in order to carry out the design, development and evolution of other workshops.

Overall, an interactive system to support the work practice in a given application domain is not a monolithic piece of software but it is developed as a network of system and application workshops, each one specific for a community of users. In other words, application workshops are those used by the different end user sub-communities to perform the tasks in their work practice, while system workshop are those used by the different experts in the design team to perform design activities. Differently from

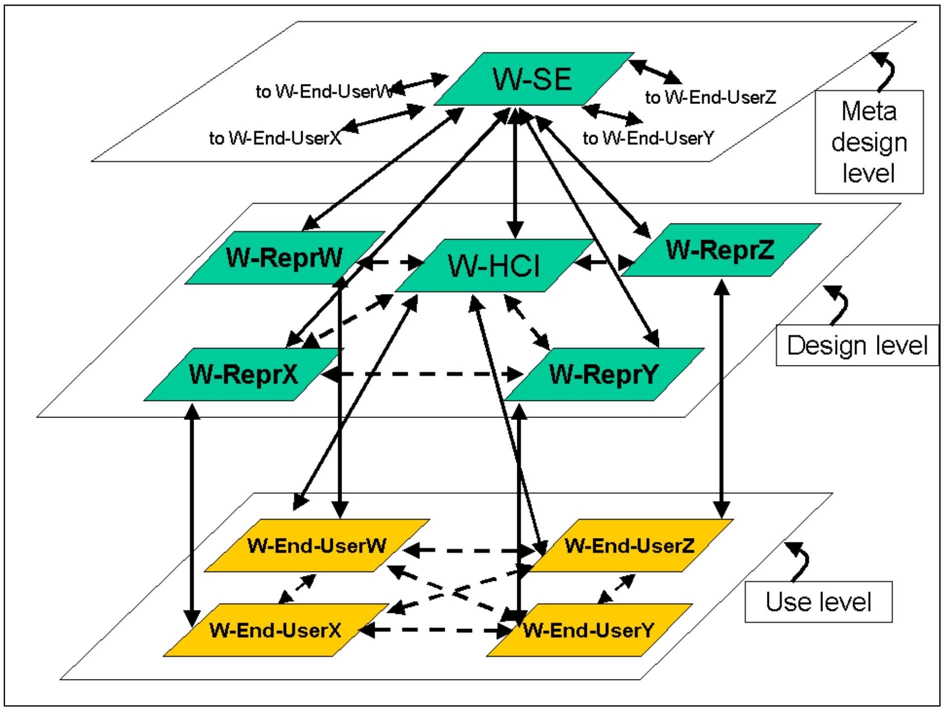


**Figure 3. A general SSW network (dashed arrows indicate exchange paths, plain arrows indicate request and generation paths).**

software engineers, domain experts and HCI experts perform design activities (creating, evaluating and modifying workshops) through direct manipulation, without the need of knowing any programming language.

## 4.1 The Software Shaping Workshop network

The SSW network is structured so that the different stakeholders can participate in the application workshops design, implementation, use and co-evolution. Every stakeholder in the design team can access more than one workshop. For example, a domain expert performing her/his working activities acts as an end user and uses an application workshop. The same domain expert acting as a designer uses a system workshop. A software engineer uses a particular system workshop to produce the tools for the other members of the team, and can access and use every other workshop to check its functionalities.

In general, a network is organized in levels. At each level there are one or more workshops, which are connected by communication paths. A generic workshop network includes three levels (Figure 3):

a) the *meta-design level* (the top level), where software engineers use a system workshop (W-SE in Figure 3), to shape the tools to be used in the next level and to participate in the design, implementation, and validation activities of all the workshops in the network;

b) the *design level*, where HCI experts and domain experts cooperate to the design, implementation, and validation activities of application workshops supporting users belonging to the different sub-communities of a given community; in this level there are specific system workshops for representatives of end users (W-ReprX, W-ReprY, W-ReprW, and W-ReprZ in Figure 3) and for HCI experts (W-HCI in Figure 3);

c) the *use level*, where end users of the different sub-communities cooperate to achieve a task: for example, in Figure 3, end users belonging to the sub-community X participate in the task achievement using the application workshop W-End-UserX customized to their needs, culture, and skills.

On the whole, both meta-design and design levels include all system workshops that support the design team in their activities of participatory design.

## 4.2 Supporting the co-evolution process

As discussed in previous sections, co-evolution results into a cyclic process, in which system usage induces an evolution in the user knowledge, culture and socio-organizational contexts, which in turn induces the evolution of the system functionalities and possibly some changes in the technology on which the system is based. Tailoring and customization tools facilitate this process, allowing end users to be active partners in it. The co-evolution process never ends until the interactive system is in use, and it encompasses all stakeholders in the network, along with all application and system workshops used by them.

In order to favor the co-evolution process in Figure 2, communication paths must be guaranteed among application and system workshops. In this way, once the overall interactive system (all workshops of a network) is in use, the design team has the possibility to observe end user activities, the new usages of the system, the new procedures induced by the evolving organization. Thanks to the communication possibilities the network offers, the design team also receives end user complaints and suggestions about the workshops they interact with. On the basis of these observations, the design team updates the system and sometimes also the underlying software technologies. In this way, co-evolution is supported.

More specifically, at the use level, end users exchange data related with their current task, to achieve a common goal. At the design level, HCI experts and domain experts exchange data and programs specifying workshops. HCI experts and domain experts also communicate with software engineers when it is necessary to forge new tools for their activities. Moreover, requests for workshop modification or extension can be sent to the design level or to the meta-design level from the lower one. Finally, when new tools or workshops are created at high levels, they are made available to the lower ones. Precisely, communication paths can be classified as:

1) *exchange* paths: they are the paths along which the exchanges of data and programs occur. Exchange paths are those existing among the workshops at the same level;
2) *request* paths: they are concerned with the communications going from low levels to higher levels; these communications trigger the co-evolution process, carrying on the feedback from end users that may include requests for workshop modification or extension;
3) *generation* paths: they represent the activity of using system workshops at a high level to generate, modify or extend workshops to be used at the lower level; new or evolved workshops are made available to lower levels along such generation paths.

At the meta-design level, software engineers produce the initial programs implementing system workshops used at the design level by HCI experts and domain experts. Interacting with their system workshops, the members of the design team produce the initial programs implementing application workshops used at the use level. According to the co-evolution process, the interaction with application workshops and system workshops determines the arising of new needs for end users and members of the design team. Such needs may include either the improvement of workshop usability as well as the request of new interaction possibilities and/or new functionalities. Request paths are crucial to allow an end user or a designer to notify her/his problems or requests to the higher level. In particular, an end user or a designer finding problems during her/his interaction with a workshop, has the possibility to annotate such problems in the workshop itself. The problems might depend on either lack of functionalities or poor usability. Annotations can be made available to all the experts reachable in the network along the request paths. The experts analyze these annotations, communicate among them using exchange paths (or request paths, if they in turn refer to the higher level), and agree on a possible solution to the notified problems, thus updating the corresponding workshop. Co-evolution is thus the result of a combination of generation, request and exchange activities that are carried out throughout the lifecycle of the SSW network.

In principle the SSW lifecycle never ends, but software engineers would be required for software changes more rarely than in current practice. In fact, conceiving an interactive system as a network of software environments (the workshops) permits to transfer as much as possible the responsibility of system evolution to the customer, who does not only acquire application workshops to be used by end users, but also buys the software (system workshops) to support co-evolution processes.

## 5. CASE STUDY

We are currently working to a project with the physicians of the neurology department of the "Giovanni XXIII" Paediatrical Hospital of Bari. In this project, different communities of physicians are involved, namely neurologists and neuro-radiologists, in the analysis of clinical cases and in the generation of the diagnosis.

We have performed a field study to identify the ambient and organization factors influencing the work of the physicians, the flow of activities, and to study the end users involved and their main tasks.

We observed that, in serious and difficult cases, physicians exchange consultation with other physicians to better study the pathology of the patient. For example, the neurologist might refer to a neuro-radiologist for a more detailed analysis of the patient Magnetic Resonance Images (MRIs) and the neuro-radiologist may need to consult another neuro-radiologist specialized in that particular pathology.

Currently, consultations occur during a real meeting in the neurology department where consultants may come from other hospitals. The cases are discussed one at a time and always with the same procedure: a neurologist chooses a case, describe the most relevant data about the clinical history of the patient and gives the MRIs to the neuro-radiologist. The neuro-radiologist chooses 3 or 4 images of interest and put them on the diaphanoscope to analyze them. Neurologists and neuro-radiologists exchange information in order to clarify possible doubts and converge to an agreed opinion. At the end, the diagnosis on which the specialists agreed is written on the patient

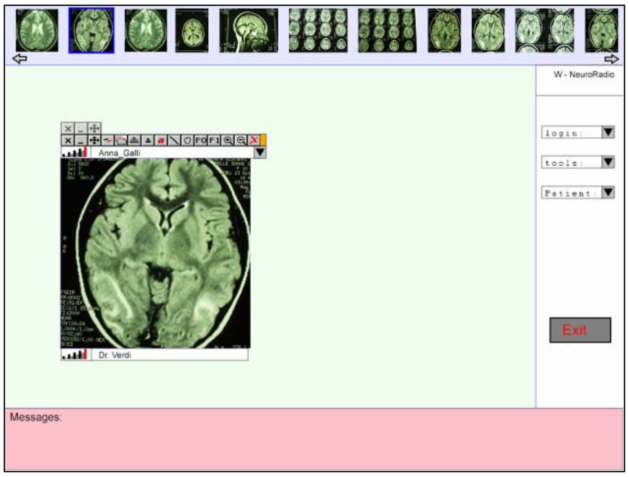


**Figure 4. An application workshop devoted to neuro-radiologists.**

record and the next clinical case is considered.

The requirements identified during the field study are at the basis of the design of the application workshops to be used by the two different communities of physicians. Figure 4 shows a screenshot of the W-NeuroRadio application workshop devoted to neuro-radiologists. It presents an overview area on the top of the screen, which is used to browse MRIs. The overview area is the electronic counterpart of the diaphanoscope used by the physicians in a real meeting. Besides the overview area, W-NeuroRadio presents a large working area, an archive area on the right side, and a message area at the bottom to show messages explaining the meaning of the elements of the workshop. In particular, the archive area hosts repositories for choosing the login type (e.g. senior or practicing physician), the tool for cooperative work (such as "consultation request") and the patient to be studied. Whenever an element in the overview area is selected, its larger image is shown in a window that, according to the "artisan's workshop" metaphor, is called *work bench* and is equipped with the tools necessary to work on the image. In particular, it provides a tool for tracing closed curves so that the neuro-radiologist can circle areas of interest to be discussed with her/his colleagues. Tools for formulating requests of consultations are also provided.

During the field study, we noticed that while neuro-radiologists are only interested in MRIs when studying neurology diseases, neurologists primarily study a great number of Electroencephalographies (EEGs) but, in some cases, they analyze MRIs. Thus, in their application workshop there are two overview areas to browse MRIs and EEGs respectively. They are resizable and the neurologists can reduce (or even close completely) the area containing the MRIs in order to expand the EEG area according to their needs. Figure 5 shows a screenshot of W-Neurologist, the application workshop devoted to neurologists. A portion of EEG has been selected from the EEG overview area and automatically loaded in one of the two work benches appearing in the working area. Each work bench is customized to the task to be performed (examining EEGs or MRIs) and to the users' needs.

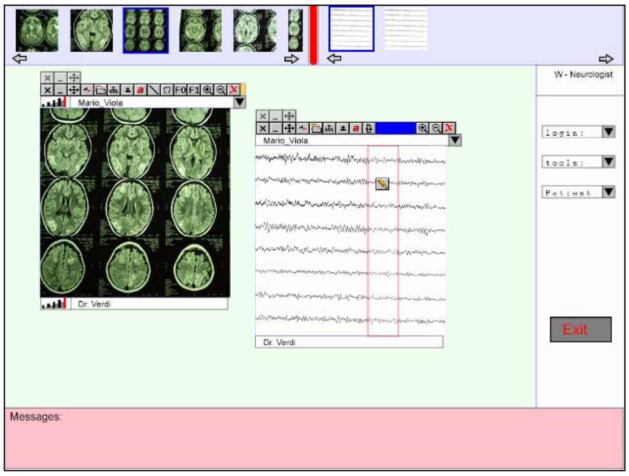


**Figure 5. The application workshop prototype devoted to the neurologist.**

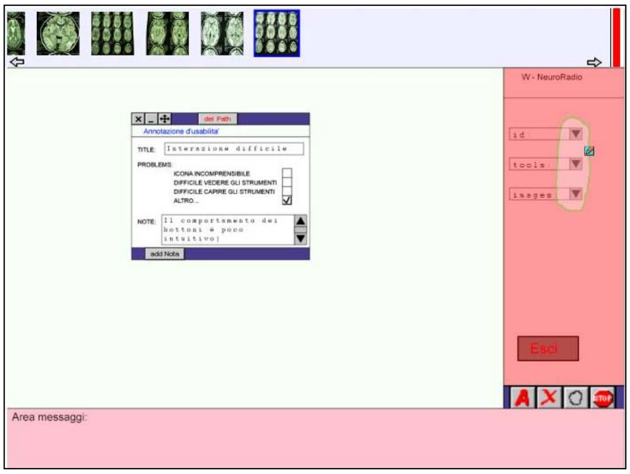
## 5.1 The SSW network in the case study

The SSW network for the analyzed domain depends on the work organization. To carry out diagnostic activities, at the use level, neurologists and neuro-radiologists use application workshops, W-Neurologist and W-NeuroRadio respectively, tailored to their culture, skills, and articulatory abilities. At this level, neurologists and neuro-radiologists exchange data (images, textual or visual annotations) with the aim of achieving a common diagnosis.

Neurologists and neuro-radiologists are experts that possess the knowledge about the domain and about the language and notation of the specific community they belong to. In our approach, they are involved in the design team. Specifically at the design level, representatives of them (domain experts) are provided with customized system workshops to be used to generate and maintain the application workshops. As in the general SSW network discussed above, a further system workshop, W-HCI, is used by HCI experts to check the human factor aspects of the generated workshops. HCI experts and domain experts collaborate with software engineers (who use the system workshop W-SE at the meta-design level) in an asynchronous and distributed participatory design, bringing their own knowledge and expertise to create and evolve the two application workshops used by the two communities of end users (neurologists and neuro-radiologists).

## 5.2 Co-evolution of users and systems in the case study

As described in Section 4.2, communications occurring along the request paths may trigger co-evolution processes. For example, at the design level, domain or HCI experts can send to software engineers requests for developing new tools to be used in their workshops; whilst at the use level, end users can communicate to HCI experts or domain experts their usability problems, which may concern the meaning of some data representations or the use of some tools. End users can also ask for workshop extensions or modifications to support emerging needs in their work context or in the tasks to be accomplished. To this aim, users interacting with a system or an application workshop may add textual or graphical annotations about usability problems or may explicitly ask for new tools or interaction possibilities to be added by switching the workshop into the so-called "annotation mode". Figure 6 shows a screenshot taken during the annotation of W-NeuroRadio by a neuro-radiologist. The user has selected "annotation mode" from the "tools" menu in the archive area. A button panel appears at the right-bottom corner, providing tools that permit to insert textual annotations, delete annotations, highlight an area of interest by circling it, and stop the annotation mode. In Figure 6 the neuro-radiologist has selected the third button in such a botton panel and has highlighted the portion of the interface that creates some usability problems.

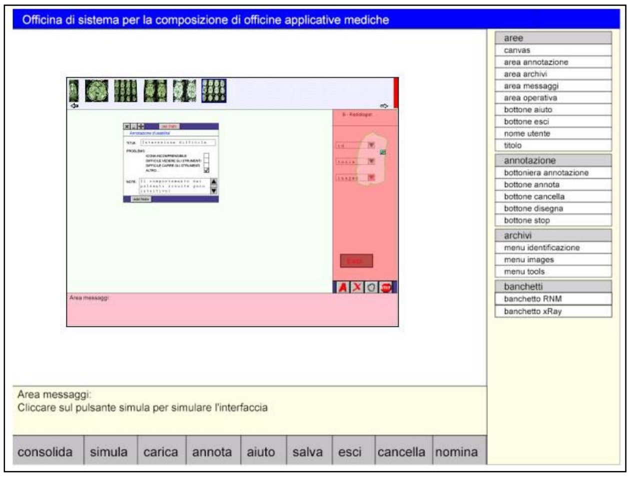


TITLE: Interazione difficile

PROBLEMS:
ICONA INCOMPRENSIBILE
DIFFICILE VEDERE GLI STRUMENTI
DIFFICILE CAPIRE GLI STRUMENTI
ALTRO... ✓

NOTE: Il comportamento dei bottoni è poco intuitivo

**Figure 6. W-NeuroRadio is in the annotation mode and the user is interacting with the annotation window after identified the three buttons thorough a curve.**



Area messaggi:
Cliccare sul pulsante simula per simulare l'interfaccia

**Figure 7. The domain expert is interacting with his system workshop to access the annotated application workshop W-NeuroRadio.**

Then, the user explains the problem in the annotation window appearing in the work area (see Figure 6). In this window, the user can type a title that synthetically describes the problem and select one or more check boxes specifying the kind of problem (e.g., the icon is not understandable, the tool is not visible, etc.); the user may also add a note describing the problem in more details. In the example, the problem is that the behavior of the identified buttons is not intuitive, because such buttons only permit to open and close the related menus.

By exploiting the request path along the workshop network, the annotated workshop is then made available to the designers. Figure 7 shows the screenshot of the system workshop used by a senior neuro-radiologist (domain expert) that is accessing the annotated application workshop.

Domain experts and HCI experts may directly modify the application workshop to solve the user problem or they may require the intervention of software engineers. In the latter case, domain and/or HCI experts can add their own annotations to the user annotated workshop: for example, they may require that a different behavior is implemented for the buttons creating the problem. To satisfy this request, software engineers probably will redesign the widget. The new widget is then provided to the system workshops at the design level, where it is possible to modify the application workshops by substituting the old widget with a new one.

All workshops in the networks may be potentially involved in the co-evolution process. In fact, on one hand, various application workshops may either be affected by the same usability problem or need to be extended with a same new functionality (in the medical case, it could be a vocal support to the interaction, a new tool for image processing, etc.); therefore, all such application workshops may be evolved. On the other hand, system workshops are evolved whenever new tools need to be designed. There are also cases that force software engineers to evolve. As an example, let us suppose that vocal support to the interaction, which is required by end users, cannot be implemented in the existing workshops for technological reasons. Software engineers are then forced to search for a more proper technology and apply it in order to provide new features in the existing workshops.

## 6. FINAL REMARKS

Novel design methodologies supporting co-evolution were investigated and discussed in [5][8]. Carroll and Rosson focused on the task/artefact loop, whilst Bourguin et al. extended this model observing that co-evolution does involve also organizational contexts and technological changes. With SSWs we support a smooth and continuous co-evolution, which does not only involve the end users and the systems they interact with, but also the design team (including representatives of end users, HCI experts, software engineers), the systems and tools used by the members of the design team, the organizational context and the technology. To this end, we adopt a meta-design perspective [10][13]. Our approach has some similarities with the work described in [6], where a system is first created by meta-designers (software engineers) to be used by caregivers who design and create scripts supporting people with cognitive disabilities. As in our approach, particular attention is put on the kinds of users involved in the domain, and tailored environments are provided to them. Co-evolution may also be activated since tools for automatic feedback and remote observations are used to notify problems to the design team.

While in this paper co-evolution is mainly triggered by users' explicit requests, we are also working to create systems that, by observing user interactions and extracting interaction patterns, can infer the need of co-evolution to satisfy new users' needs or overcome usability problems. The work in [2] is in this direction: a multi-agent system running in background with the interactive system is able to capture user actions and system reactions, and recognize user preferences during the interaction, and anomalous or inefficient interactions. We believe that an automatic support is useful to activate the co-evolution process, thus we are integrating it in the SSW approach.

The software prototypes presented in Section 5 are currently under evaluation with neurologists and neuro-radiologists, who will give their opinion about the appropriateness of the tools for tailoring and co-evolution we are developing. The feasibility of the SSW approach has also been demonstrated in mechanical engineering [9] and geology [7] domains. The adoption of the SSW approach in such domains results in much less effort in system maintenance and in greater user satisfaction.

## 7. ACKNOWLEDGEMENTS

The authors wish to thank the physicians of the Hospital “Giovanni XXIII” in Bari for their collaboration. They are also grateful to Piero Mussio for the stimulating discussions during the development of this work.

## 8. REFERENCES

[1] Abowd, G. D., Beale, R., Users, systems and interfaces: a unifying framework for interaction. In Diaper, D., Hammond, N., eds., *HCI'91: People and Computers VI*, pp. 73-87. Cambridge University Press, Cambridge, 1991.

[2] Arondi, S., Baroni, P., Fogli, D., Mussio, P. Supporting co-evolution of users and systems by the recognition of Interaction Patterns. *Proceedings of the International Conference on Advanced Visual Interfaces (AVI 2002),* Trento (I), May 2002, 177-189.

[3] Berners-Lee, T., What the Semantic Web can represent, 1998, http://www.w3.org/DesignIssues/RDFnot.html.

[4] Bottoni, P., Costabile, M.F., and Mussio, P., Specification and Dialogue Control of Visual Interaction through Visual Rewriting Systems, *ACM TOPLAS*, 21-6, 1999, 1077-1136.

[5] Bourguin, G., Derycke, A., Tarby, J.C. Beyond the Interface: Co-evolution inside Interactive Systems - A Proposal Founded on Activity Theory, *Proc. IHM-HCI 2001*.

[6] Carmien, S., Dawe, M., Fischer, G., Gorman, A., Kintsch, A., Sullivan, J. F., Socio-Technical Environments Supporting People with Cognitive Disabilities Using Public Transportation, *ACM Trans. on CHI*, 12(2), 2005, 233-262.

[7] Carrara, P., Fogli, D., Fresta, G., Mussio, P. Toward overcoming culture, skill and situation hurdles in human-computer interaction. *Int. Journal Universal Access in the Information Society*, 1(4), 2002, 288-304.

[8] Carroll, J.M., Rosson, M.B., Deliberated Evolution: Stalking the View Matcher in design space. *Human-Computer Interaction*, 6 (3 and 4), 1992, 281-318.

[9] Costabile, M.F., Fogli, D., Fresta, G., Mussio, P., Piccinno, A., Software Environments for End-User Development and Tailoring, *Psychnology*, 2(1), 2004, 99-122.

[10] Costabile, M.F., Fogli, D., Mussio, P., Piccinno, A., A meta-design approach to End-User Development, *IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC'05*, Dallas, Texas, USA September 20-24, 2005, 308-310.

[11] Costabile, M.F., Fogli, D., Mussio, P., and Piccinno, A., End-User Development: the Software Shaping Workshop Approach, to appear in Lieberman, H., Paternò, F., Wulf, V. (Eds), *End User Development - Empowering people to flexibly employ advanced information and communication technology*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2006.

[12] Dillon, A., Watson, C., User analysis in HCI: the historical lesson from individual differences research, *International Journal of Human-Computer Studies* 1996, 45(6), 619-637.

[13] Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A.G., Mehandjiev, N., Meta-Design: a Manifesto For End-User Development, *Communications of the ACM*, Vol. 47(9), Sept. 2004, 33-37.

[14] Hutchins, E.L, Hollan, J. D, Norman, D., Direct manipulation interfaces, *User Centred System Design*, Norman D. and Draper S., eds., Hillsdale, NJ: Lawrence Erlbaum Associates, 1986, 87-124.

[15] Nielsen, J., *Usability Engineering*, Academic Press, San Diego, 1993.

[16] Nielsen, J., Mack, R.L., Bergendorff, K.H., Grishkowsky, N.L., Integrated software in the professional work environment: evidence from questionnaires and interviews. *Proc. CHI 86 Conf.,* Boston, MA, 1986, 11-12.

[17] Norman, D. A, Human-Centered Design Considered Harmful. *Interactions of ACM*, 12(4), (July), 2005, 14-19.

[18] Schuler, D., Namioka, A., Preface, *Participatory Design, Principles and Practice*, Lawrence Erlbaum Ass. Inc. Hillsday, vii, N.J, 1993.