

Master thesis

for the MOSIG M2R
defended by

Dimitri Masson

Genetic Algorithm for Creativity Enhancement in UI design

25 Juin 2010

Tutored by Alexandre Demeure and Gaëlle Calvary

Abstract

User Interface (UI) design is an iterative and evolutionary process. First steps require creativity from the designer to produce innovative UI. While many tools have been developed to reduce the burden of the designer, few provide a support for creativity. A popular approach to computer support for creativity can be founded in Interactive Genetic Algorithms (IGA). They have been applied with success in many creative activities (e.g. painting). However, works based on this approach had little incidence in the HCI community, possibly due to the poor quality of the results. This work proposes a new approach that combines IGA with model based UI. Magellan, [Masson 2010] a prototype of this approach had permitted to assess the potential of the approach through qualitative evaluation.

Keywords: Creativity, Interactive genetic algorithm, Model based User Interface.

Resumé

La conception d'interfaces utilisateur suit un processus itératif. Le concepteur doit faire preuve de créativité dans les premières étapes de la conception. Si de nombreux outils existent pour lui faciliter la tâche, peu fournissent un soutien à la créativité. Une approche répandue pour l'assistance à la créativité par ordinateur est l'utilisation d'algorithmes génétiques interactifs (IGA). Cette approche a été utilisée avec succès dans de nombreuses activités impliquant de la créativité (ex: la peinture). Cependant, les travaux qui s'appuient sur cette approche n'ont eu que peu d'impact en IHM. La pauvreté des interfaces produites pourrait en être une explication. Ce rapport présente une nouvelle approche combinant les IGA et les interfaces utilisateurs basées sur modèles ainsi que Magellan [Masson 2010], un prototype qui a permis de valider l'approche au cours d'une évaluation qualitative.

Mots clefs: Créativité, Algorithmes génétiques interactifs, Interface utilisateur basée modèles.

Sommaire

- Abstract..... 2
- Sommaire 3
- 1. Introduction..... 5
 - Problem Statement 6
 - Approach..... 6
 - Outline..... 7
- 2. State of the art 8
 - User Interface Design 8
 - Tools 10
 - Sketching tools 10
 - Prototyping tools..... 12
 - UI toolkit and interface Builder 14
 - Model Based tools..... 15
 - Creativity 17
 - Creative Product..... 17
 - Creative Person..... 18
 - Creative Process 18
 - Creative evolutionary systems in general 19
 - Creative evolutionary systems for UI design..... 21
 - Conclusion of the state of the art 24
- 3. Contribution 26
 - Running Case study..... 26
 - Overview of the approach 26
 - Genetic Representation of Individuals 27
 - Individuals encoding..... 28
 - Initialization 29
 - Mutation 29
 - Node mutations 30

Edge mutation	31
Inbreeding	32
Fitness Evaluation.....	32
Selection of the individuals	33
Flexibility of the process.....	33
Prototype.....	34
Magellan UI	34
4. Evaluation of the approach	37
Semi-directed interviews.....	37
Experimental protocol	37
Interview synthesis.....	39
Creative Process	39
Constraints	40
Inspiration.....	41
Innovative elements	41
Magellan.....	42
5. Conclusion and future works.....	44
6. Bibliographie.....	46

1. Introduction

The User interface (UI) is an important part of Interactive System (IS). It makes it possible for the end-users to use the functions provided by the IS. The development of the UI represents an important part of the whole IS development. According to [Seffah et al. 2005], the code of the UI represents more than 50% of the total code of the IS. Hence, from the programmer point of view, getting a suitable UI design as soon as possible is crucial to avoid the increasing cost when changes are done later (Figure 1). [Bias 1994] found that once a system is under development, correcting a problem costs 10 times as much as fixing the same problem at design time. If the system has been released, it costs 100 times as much relative to fixing at design time.

From the end user point of view, a poorly designed UI can result in inefficient uses and can lead to bugs. [Boehm 1991] reported that more than 80% of reported bugs were due to bad UI designs.

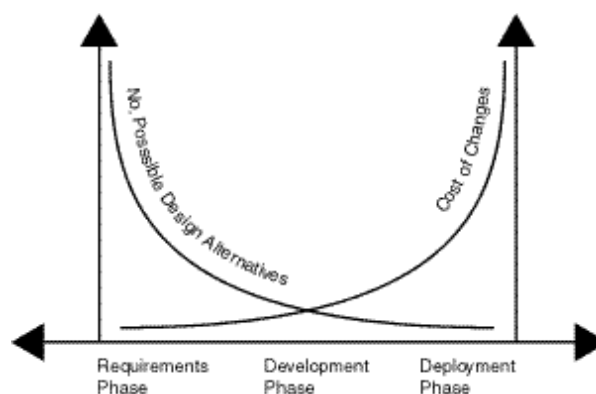


Figure 1: The number of possible designs decreases as the cost to make changes increases. Quoted from [Bias 1994], p. 80.

Generally speaking, design can be seen as a purposeful, constrained decision making activity. It implies a set of variables which values have to be decided to determine a particular design. This set is referred to as the design Space [Gero 1996]. This design space is often large and likely to evolve along the design process as new constraints emerge, resulting in the introduction of new variables. [Buxton 2007] and [Tohidi 2006a] identify two overlapping steps in the design process: "Getting the right design" and "getting the design right". Getting the right design implies elaborating numerous designs, each exploring opportunities. As they are numerous, designs at this step of the process must be cheap to produce. The goal is to suggest solutions rather than formally describe them, letting place for broad interpretations. The next step of the design process consist in selecting promising designs and narrowing the possible interpretation by making decisions (Figure 2) until a final, implemented, design is reached (focal point on Figure 2).

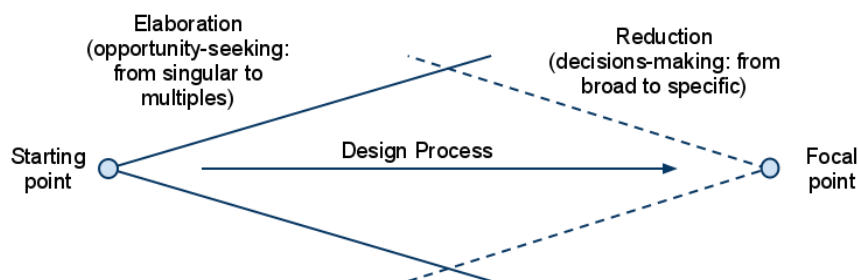


Figure 2: The design process is composed of two overlapping tendencies: One elaborating new designs, the other selecting and refining them. Quoted from [Buxton 2007], p144.

A lot of tools are provided to UI designers to support them in alleviating the specification of the UI. One can cite builders that allow to visually construct the UI or sketching tools, such as SketchiXML [Coyette 2004], that allow to build low fidelity to high fidelity prototypes of the UI. On the contrary, very few works have been done to support designers in exploring the design space.

Problem Statement

In this work, I tackle the problem of supporting the designers in getting the right UI design along the design process of an IS. I do not aim at fostering the design process of the whole IS. Instead, I focus on the design process of the UI only. I assume that the functionalities of the IS (*i.e.* what the users can do with the system) are already defined. I focus on the design of the UI part (*i.e.* how the users can use the system).

My goal is to provide designers with a tool that help designers to explore the design space and offer a source of inspiration. Helping designers to get inspired and to produce numerous UI designs should also help designers to identify the right UI design early on and therefore save a lot of resources as many design flaws can be spot in early stages of the design.

Even for a simple IS, the design space is infinite. Indeed, it exists an infinity of possibilities to represent the tasks users can do (*e.g.* "specifying a text" can be represented by a text entry, a free writing area, speech recognition, etc.) and an infinity of possibilities to give users access to them (*e.g.* aligning element in column, in a 3D space, using tabs, menus, etc.). However, most of the possible designs are irrelevant because either representations of user tasks or the way to assemble them are irrelevant (Figure 3).

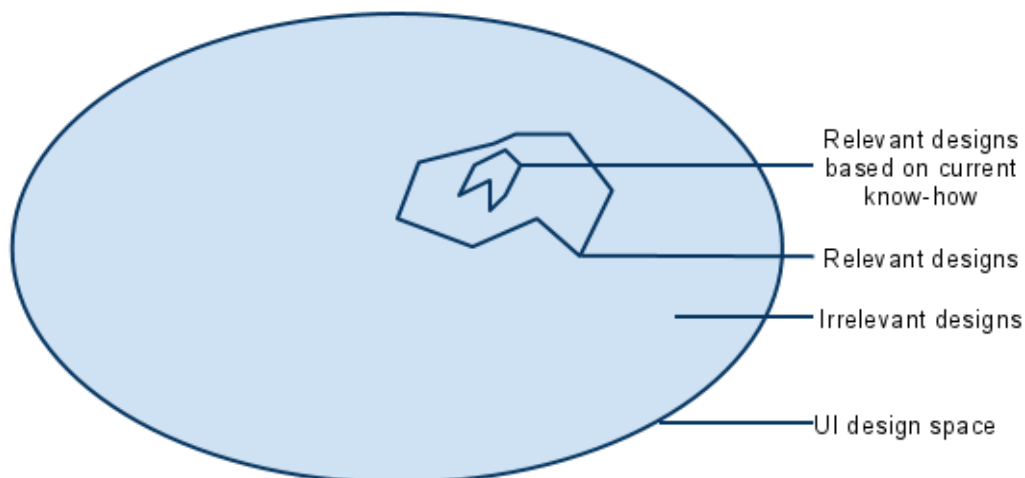


Figure 3: The UI design space is mostly constituted of irrelevant designs. Among the relevant designs, only a subset can be addressed by the current know-how.

As a consequence, I choose to consider a subspace of the design space where both the way to represent user tasks and to assemble representations come from the know-how of the UI designers' community (Figure 3). Such a know-how is one of the constituent of the model-based approaches in HCI.

Approach

My approach relies on two domains: Model based UI and Interactive Genetic Algorithms.

Model based UI approaches rely on models to describe UI at different levels of abstraction. In particular, the task and concepts model describes the tasks that users can do and the concepts they manipulate independently of the way these tasks will be done. Such a high level description can be

transformed into much concrete ones until a final UI is produced (e.g. HTML code). Transformations between these models express how descriptions of one level should be translated into another (e.g. a “choose an item” task is transformed into a “list of radiobuttons”). These transformations do contain a know-how of the HCI community and taking it into account is something important in order to provide relevant designs. As a consequence, I propose to explore the use of task and concept models coupled with transformations expressing the know-how of UI designers as a basis for generating relevant UI designs.

In order to enable designers to navigate in the relevant design subspace, I propose to use Genetic Algorithms (GA). GA are a search technique inspired by Darwinian theory. They apply to optimization problems that have no analytical solution but in which the quality of potential solution can be evaluated. GA are implemented in a computer simulation in which a population of abstract representations (called chromosomes or the genotype of the genome) of candidate solutions (called individuals, creatures, or phenotypes) evolves toward better solutions. The evolution starts with an initial population, usually randomly selected. Individuals are then evaluated, best individuals are retained to be part of a mating pool to form a new population. Stochastic mutations are applied to the new population before the cycle starts over. GA work well on large space problem. Interactive Genetic Algorithms (IGA) are a special case of GA where the fitness function which evaluates the quality of an individual is replaced by a human evaluation. IGA combine the power of GA with the human expertise. They have been applied on problem for which no computational function does exist to evaluate individuals, such as artistic problems [Sims 1991, Secretan 2008].

In summary, the approach I consider in this work is based on the combination of Model Based UI design and IGA to foster the exploration of the relevant design subspace and to product relevant UI designs.

Outline

Section 2 presents a state of the art in three related fields: User Interface Design, Creativity and Creative Evolutionary System. Section 3 presents my contribution including Magellan [Masson 2010] a first prototype. Section 4 elaborates on the benefits of using IGA for UI design. Finally, section 5 concludes and presents the future works.

2. State of the art

In this section, I first present an overview of the typical User Interface Design process in order to position my problem this process. I then describe the different sorts of tools that can be used by designer to build UI and their appropriateness with respect to the design process. The next part is dedicated to an overview of the concept of creativity before concluding by a state of the art of creative evolutionary systems and their application to UI design.

User Interface Design

Modern theories on UI design, including User-Centered Software Design [Seffah 2005], identify UI development as an evolutionary, iterative and incremental process. A general cycle of UIs development as presented in Figure 4 consists of three steps: Problem Preparation, Ideation and then Evaluation.

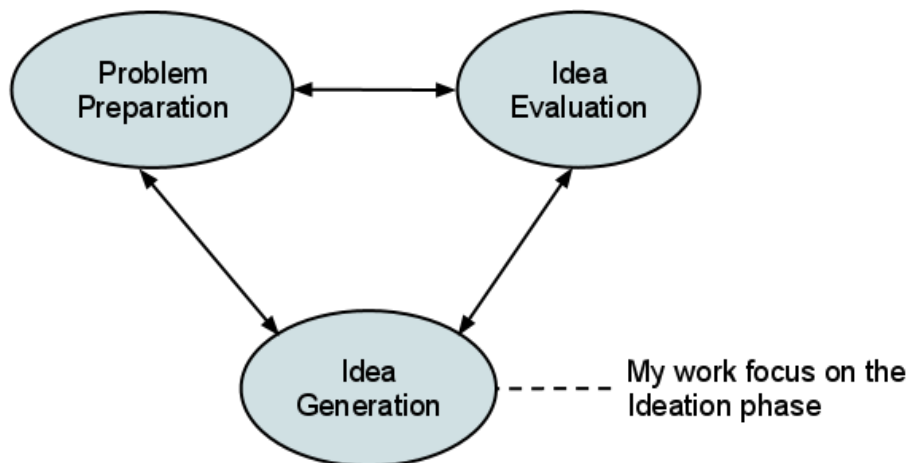


Figure 4: Overview of the design process [Warr 2005].

The design process starts with a problem preparation step in which designers imagine various scenarios and explore a large amount of concepts. The more creative is the designer, the more numerous and the richer become the scenarios. From these scenarios and from these concepts, designers identify the main characteristics of potential solutions. In particular they might identify a set of constraints, a set of objects and the tasks that are achievable. These sets may be updated at each cycle. At the end of this stage the designer get a mental image of the problem to tackle.

The second step is about exploring and comparing multiple alternative solutions to the problems identified during the previous stage of the process. Ideally, at this stage, numerous sketches of the UI are produced. Sketches are used to explore ideas and raise questions about the problem. They are tentative to suggest alternative solutions. They must not be strict answers to the problem and are intentionally ambiguous [Buxton 2007]. This step requires a high level of creativity from the designer: He must produce a myriad of variants and combination of ideas to explore the design space. My work mainly addresses this step of the process

During the Evaluation step, prototypes are set up and evaluated in context. While sketches suggest rather than describe, prototypes refine sketches and describe specific answers. In the early iterations, prototypes might be mock up. Through the iterations, they tend to the final product.

[Virzi 1996] and [Tohidi 2006a] have shown that low-fidelity prototypes are as effective as high-fidelity prototypes. They argue for using low fidelity prototypes far in the iterative process in order to reduce the evaluation cost. As low fidelity prototypes are relatively cheap to set up, it enables designers to test more concepts and provide feedbacks. This aspect is crucial for producing high quality UIs. At last, prototype evaluation provides feedback about the tested designs. This feedback enriches the designer vision of the problem, and the cycle starts again.

Buxton [Buxton 2007] presents this cycle successions in a linear manner, through a succession of expansion and refining step converging toward a final design (Figure 5). It has to be noticed that each iteration (exploratory, clarification, resolution) contains expansion and refining step. This means that the design process is not linear. Designers do not start with a set of candidate designs which is progressively reduced toward a final design. On the contrary, candidate solutions (sketches, prototypes or even quasi final designs) do appear and disappear all along the process. The reasons why the choices are made of removing, refining or adding a candidate design is as much important as the choices themselves. Therefore, designers should explicit the design rational to be able to explain the choices made in the Controlled Convergence steps.

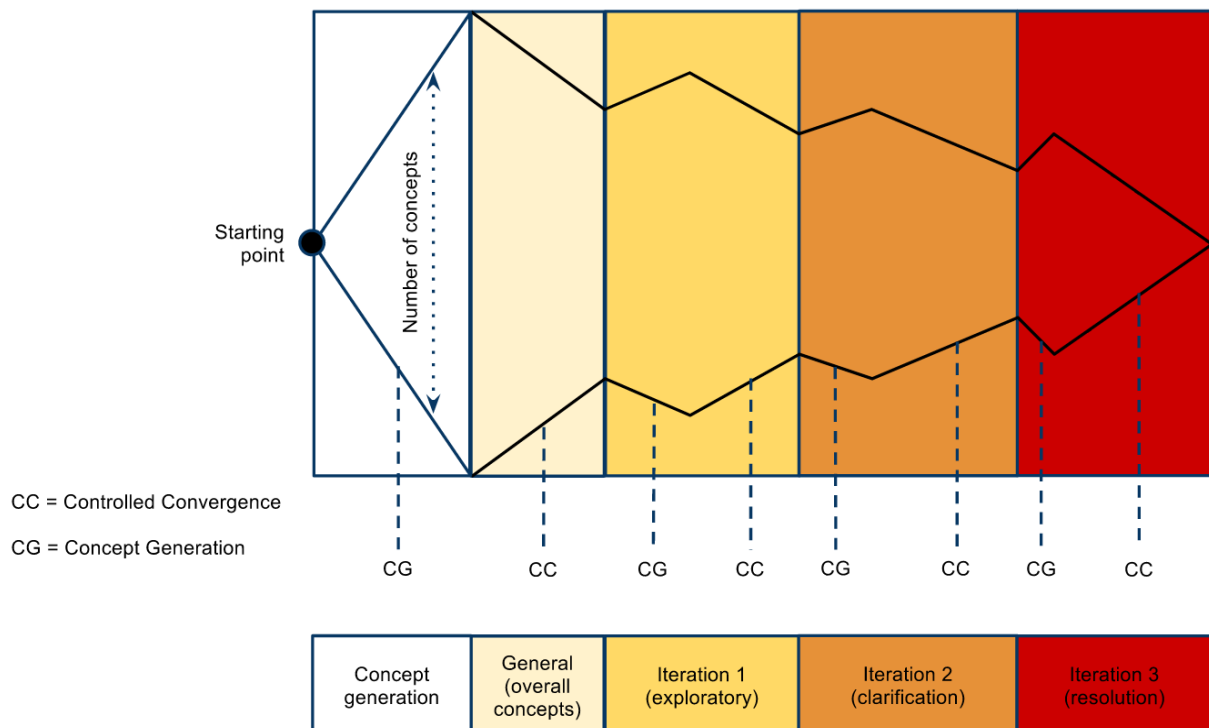



Figure 5: Concept generation and selection through an iterative process. Quoted from [Buxton 2007].

	<p>UI design process is an iterative and evolutionary process. Creativity is present at each step of this process. Each iteration is composed of expansion and reduction steps.</p>
---	---

The next subsection is focused on tools (either techniques or software) that help to design and implement the user interface.

Tools

Tools are important since they reduce the designer fatigue and allow designer to produce more quickly. Quicker steps leads to more designs' iterations. Hence, it fosters the production of high quality user interfaces. Tools are divided in four groups: Sketching tools, Rapid prototyping tools, Implementation tools and Model Based UI (MBUI). As design is a continuous process, there is often no clear distinction between sketches, prototypes and early implementation. Consequently some of the tools described below could fall in several sections.

Sketching tools

To explore a maximum number of possibilities, usage is to produce sketches. Usually, sketches refer to a simply or hastily executed drawing, especially a preliminary one, giving the essential features without the details (quoted from dictionary.reference.com). [Buxton 2007] proposed a more precise definition of sketches through a set of attributes they must fulfill. Sketches must be **quick to make** and can be **timely provided**. They are **inexpensive to build**, making it possible to explore any concept. They are also **disposable**, which means that designers can afford to throw them away when done. Indeed, the investment with a sketch is in the concept, not the execution. Sketches should be considered as a social thing, there must be **plenty** of them to be useful. Anyone should be able to recognize whether a drawing is a sketch or not, it implies using **distinct gestures** that give them a sense of openness and freedom. Sketches should only contain the **minimal details** necessary to grasp the concept they explore, they should not provide answers to questions which were not being asked. With sketches, it can be said that going beyond “good enough” is a negative rather than a positive thing. The **degree of refinement should be appropriate**, reflecting by its resolution or style the level of certainty of the designer’s mind at the time. At last, sketches are intended to **suggest and explore** concepts rather than confirm them, they are **ambiguous**.

The value of sketches lies not in the produced artifact itself (the drawing) but in its ability to provide a catalyst to the desired and appropriate behaviors, conversations and interactions. Indeed, sketches are the vehicle, not the destination: designers do not draw sketches to represent ideas that are already consolidated in their mind. Rather, they draw sketches to try out ideas, usually vague and uncertain. By examining the externalizations, designers can spot problems they may not have anticipated. Even more than that, they can see new features and relations among elements that they have drawn, ones not intended in the original sketch. These unintended discoveries promote new ideas and refine current ones [Buxton 2007].

The sketching activity can be seen as a “conversation”, or dialectic process [Goldschmidt 1991], between the designer and its sketches (Figure 6). First, sketches are created from what the current knowledge of the designer. Reading these first sketches leads the designer to interpret them in a way that (hopefully) may generate new knowledge. This extraction of new knowledge results from what Goldschmidt calls “seeing as” (bottom arrow in Figure 6). The creation (top arrow on Figure 6) results from what Goldschmidt calls “seeing that” reasoning. It means that the new knowledge is translated into sketch, which in turn leads to a new interpretations, etc.

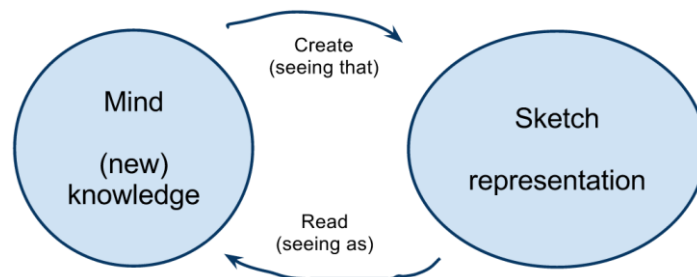


Figure 6: A sketch of a dialog with a sketch. Quoted from [Buxton 2007].

A simple yet quiet efficient tool to support sketching is a pen coupled with a sheet of paper. However, paper based sketches can hardly describe the interaction with the UI. In some case, this shortcoming can be overcome simply by using animated GIF (Figure 7). More generally, electronic tools such as SILK [Landay 1995] or DENIM [Lin 2000] have been developed to enable designers to quickly specify the interaction directly from sketches. Figure 8 illustrates how links can be expressed using DENIM to establish relationships between several sketches describing different parts of the UI. Despite the availability of pre-cited tools, there is not yet a tool that enables designers to sketch UI interaction as easily as UI presentation.

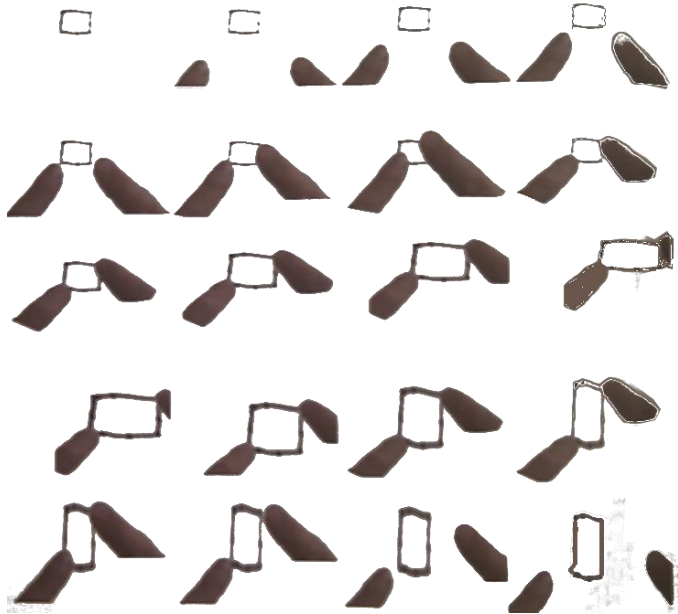


Figure 7: Animated GIF sketching how to use two fingers to resize a rectangle. Quoted from [Buxton 2007].

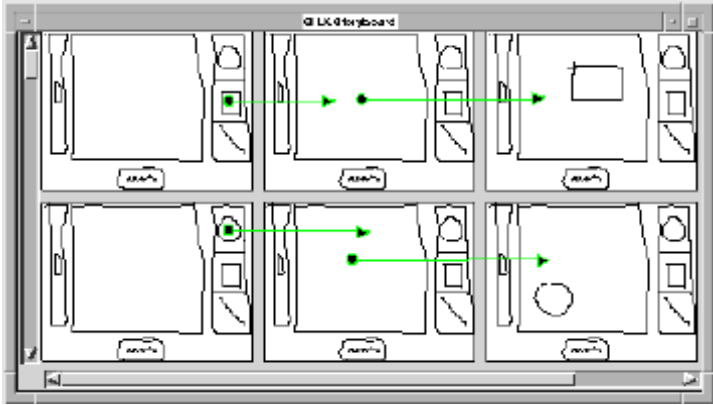


Figure 8: Specification of the interaction in DENIM [Lin 2000]. Green arrows express transitions from one sketch to another. Transitions are triggered by user click.

Sketching is sometimes compared to low fidelity prototyping as the graphical style of resulting artifacts (sketches or low fidelity prototypes) is somehow close. However, sketches and prototypes serve different purposes, and therefore are concentrated on different stages of the design process. Sketches target the early ideation stages, when concepts are generated (see first iterations on Figure 5), while prototypes target the later stages, when the controlled convergence tends to a solution (see last iteration on Figure 5). Essentially, the difference between a sketch and a prototype lies in the fact that the later is less disposable and is larger to build; therefore there are fewer prototypes than sketches. Figure 9 emphasizes that the differences between sketches and prototype takes place in a continuum (and not an either/or proposition). The purpose of building sketches or prototypes is different from

several aspects. On the overall, it could be said that sketches open new tracks [Tohidi 2006b] while prototypes test their viability.

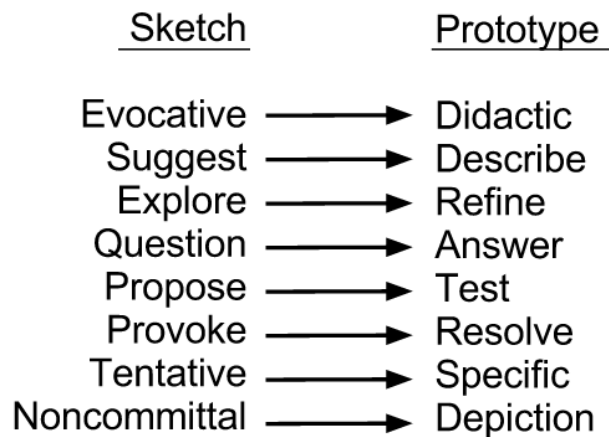



Figure 9: The Sketch to Prototype continuum. Quoted from [Buxton 2007].

	<p>Sketching is an essential part that takes place at the beginning of design.</p> <p>It helps designers in finding new ideas.</p> <p>While sketching UI presentation can be done with tools as simple as a paper and a pen, sketching UI interaction is something more complex for which no “universal” tool does exist.</p>
--	---

Prototyping tools

While sketches are used to explore possibilities, prototypes are used to test specific solutions. Prototyping also enables to acquire information needed to successfully build the IS. As a consequence, prototyping tools must ease the process of gathering such information. [Szekely 1994] identified two fundamental requirements for prototyping tools:

- 1) Accessibility and ease of use: Since designers are not programmers, prototyping tools must be accessible without intensive learning;
- 2) Extensive control and Fast turn-around: they must enable to quickly change any features of the UI and immediately display the results of these changes.

Unlike Sketches, prototypes must be executable to be effective representation of the IS. Prototype execution might reflect a faithful interaction with the IS, *i.e.* appropriate responses to user inputs. However this does not imply that the prototype might be connected to real data, and performs real computation [Szekely 1994]. For instance, simulated data and/or computation are sufficient.

Prototyping techniques such as “Wizard of Oz” can be used with profits to simulate computations of the systems depending on the real inputs of users. In Wizard of oz techniques the interactive system is replaced by a “human actuator” who reacts to real user inputs. For instance, a speech recognition based software would be really painful to prototype using programming; It can easily simulated by a human listening and operating the system. Historically, the first application of the wizard of Oz technique seems to be the airline ticket kiosk concept [Erdmann 1971]. This kiosk was supposed to understand client when they spoke. The speech recognition was done by a human “wizard of Oz”

(Figure 10). This prototype enables designers to test the concept of a speech recognition based airline ticket kiosk years before speech recognition became available on computers.

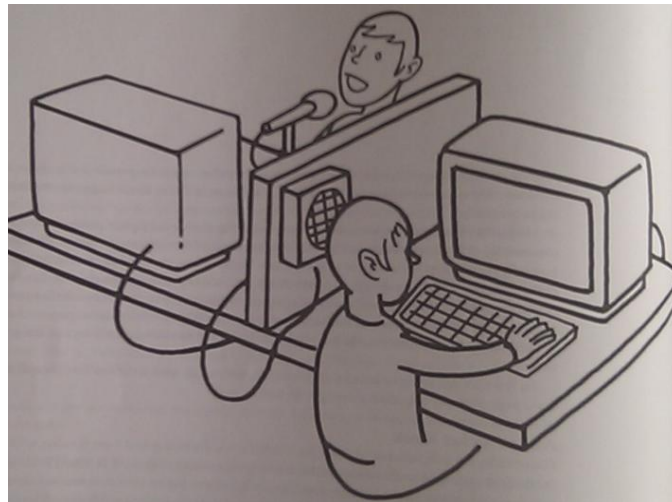


Figure 10: Schema of a prototype of airline ticket kiosk where interaction with users is done via simulated speech recognition.

Paper prototyping enables designers to simulate the IS by using sheets of paper with drawn UI (Figure 11). The users can interact with the paper by “clicking” with their fingers on UI elements. Designers are in charge of changing the sheet of paper depending on the user’s interaction. This prototyping method is useful to present early work to other designers, or to test the look and feel of the UI with users.



Figure 11: A set of sheets of paper with drawn UI can be use as a prototype.

It is sometimes useful to built higher fidelity prototypes, for example to test novel interaction techniques based on existing devices (e.g. a mouse). For instance, the ICON toolkit [Dragicevic 2004] enables designers to assemble components in a flowchart, which result in a new interaction technique (Figure 12).

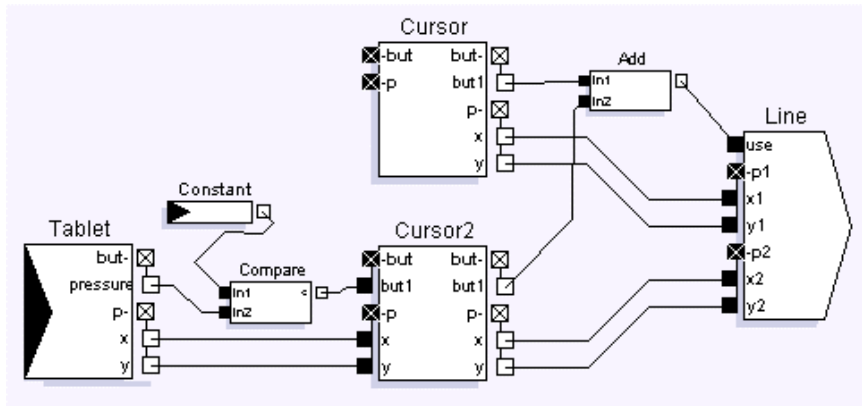


Figure 12: Example of a flowchart for prototyping an interaction technique with ICON [Dragicevic 2004].

This last example could also be used as a graphical programming tool. In the next section, we examine the tools that can be used to produce the final UI.

	<p>Prototyping enables designers to test their ideas. Prototypes can be low- to high-fidelity depending on what has to be tested. Part of the system can be simulated (Wizard of Oz technique). A lot of tools do exist to prototype UI presentation and UI interaction.</p>
--	---

UI toolkit and interface Builder

As the design cycles go, prototypes evolve toward the final product, and production tools replace prototyping techniques. At this stage of the process designers are usually replaced by programmers. Production tools aim to ease the implementation of the final user interfaces.

UI Toolkits typically provide UI components, also called widgets, to the programmer in the form of code snippets and architectural framework [Myers 2000]. Common UI toolkits are SWING, Tk, Qt. Established framework and reusable components make user interface construction easier and support interface consistency between interactive systems. For instance color picker dialog boxes is a classical component spreads between Windows application (Figure 13).

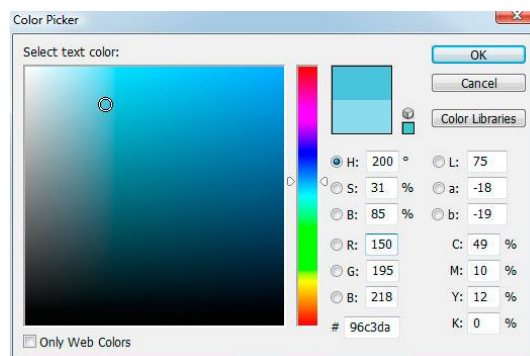


Figure 13: This color picker dialog box capitalizes a way to choose a color.

Actually, widgets are a form of capitalization of the knowledge of UI designers. Instead of reprogramming from scratch every interactive component, designers can rely on a toolbox that contains a part of the know-how of the HCI community, thus shortening the time to develop the UI and

avoiding some mistakes by reinventing the wheel. Moreover, in the case when the required component is not available in the toolkit, it is always possible to implement it.

Interface builders are a logical extension of toolkit programming. They are interactive tools that enable the programmer to visually place the UI elements and to create windows and dialog boxes. Their success is based on the use of a graphical language to deal with graphical problem (the UI element layout). These tools considerably reduce the cost of production of simple interfaces, and even make it possible to use them as prototyping tools. Current Integrated Development Environments (IDE) typically provide their interface builder, such as Visual Basic, or Borland C++ Builder. However these tools are usually limited to produce dialog and menu based interfaces, while the application specific display has to be done using conventional programming [Szekely 1993].

Model Based tools

MGUI approaches in HCI consider the UI at several levels of abstractions and the relationships existing between these levels (Figure 14). Designer uses modeling tools to produce initial models. These models can be automatically [Berti 2004], semi-automatically or even manually transformed into more concrete ones. In the case of automatic generation, design knowledge and guidelines can be taken into account to increase the quality of generated models. [Szekely 1996] envisioned that automatic design advisors could also emit critics about initial or generated designs. Finally, implementation tools and compiler can be applied from the concrete UI specification to generate the deliverable application.

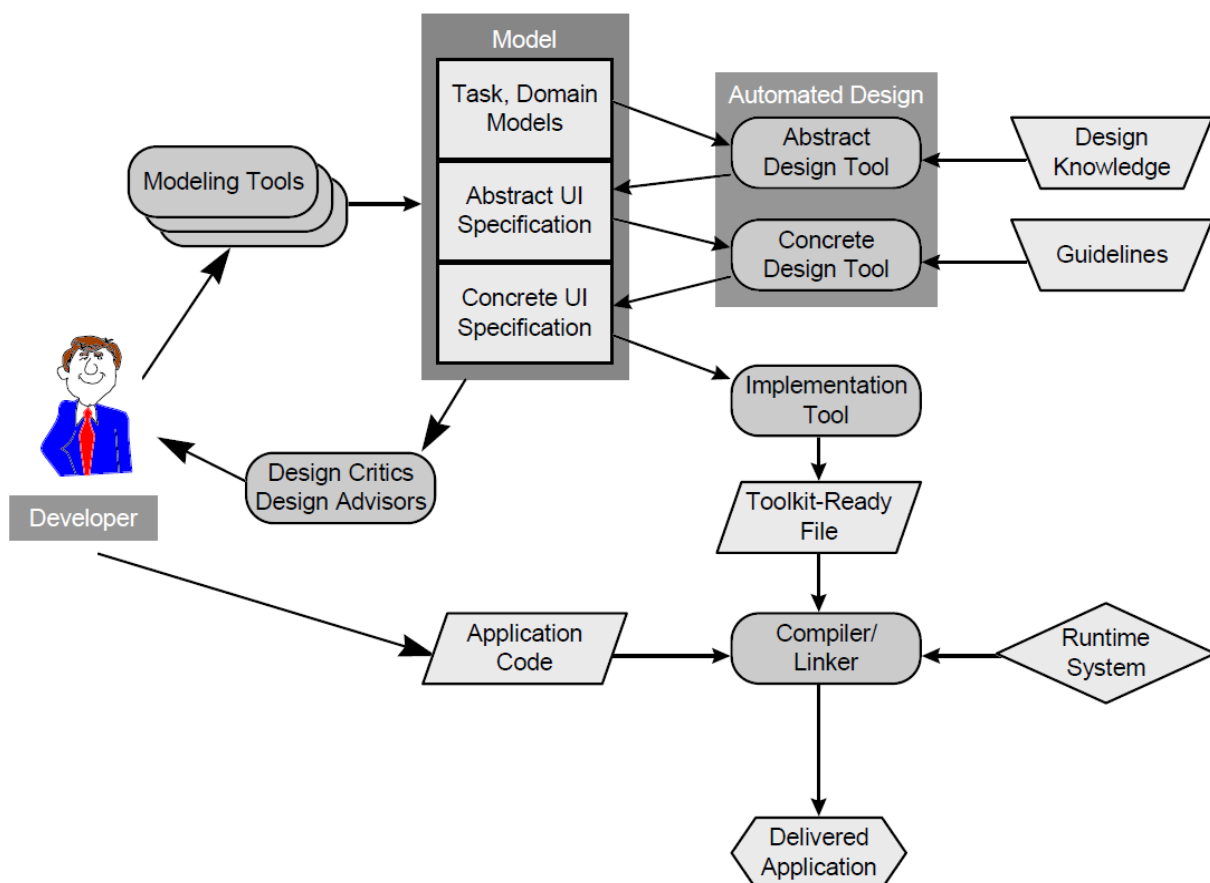


Figure 14: Model-Based Interface Development Process, quoted from [Szekely 1996].

Usually, at the higher level of abstraction, a task and concepts model describes the tasks that the users can do (e.g. editing a text) and which concept of the domain they use (e.g. the address of the user). The user task and concepts model is often represented as a tree (Figure 15). Nodes of the tree

represent user tasks. Hierarchical arcs between parent and child nodes represent how the parent task can be decomposed into subtasks. At last, relationships between sibling nodes represent task operators. Task operators express temporal constraints. For instance, the interleaving task operator (noted ||| in Figure 15) expresses that both task can be done at the same time or by switching from one to the other at any time. On the contrary, the sequence operator task (noted >> in Figure 15) expresses that the first task has to be achieved before starting the second one.

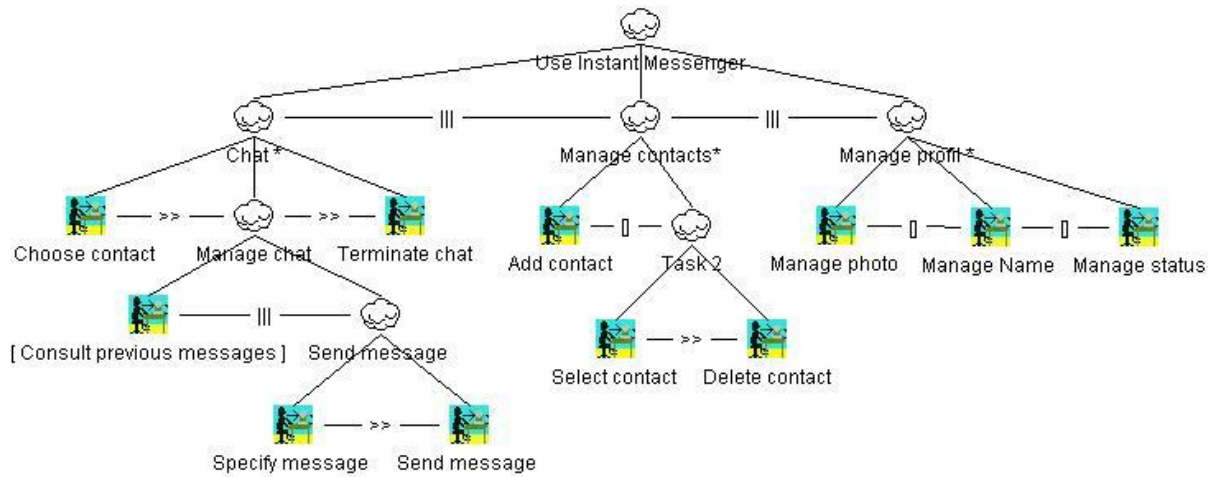


Figure 15: Example of a user task tree in CTT notation [Paterno 1997].

From this task and concept model, a cascade of transformations is applied which produce more and more concrete models (e.g. a dialog model) until one or several final UI (i.e. UI to be used by end users) are produced. The transformations applied can be automatic, semi automatic or manual and are related to different context of uses. A context of use models for what kind of users, platform (hardware and software) and environment (physical and social) the UI is supposed to take place. What is interesting for me here is that these transformations and their related context of uses somehow capitalize a know-how. Indeed, transformations do express for instance that a "choice an item" user task can be implemented by a "list of radiobuttons", or a "combobox" depending on the available screen size and/or user preferences (e.g. see the Arnault system [Gajos 2005]).

It has to be noticed that MGUI approaches can potentially cover the whole design process. Indeed, even sketches or prototypes can be used as model. Transformations linking task models, sketches and/or prototypes can express the design rational choices made during the design process.

MGUI approaches can also be used to retro-concept final user interface into more abstract level. This retro-conception can be combined with translations that transform a model to an equivalent-abstracted model but designed for another context of use (Figure 16).

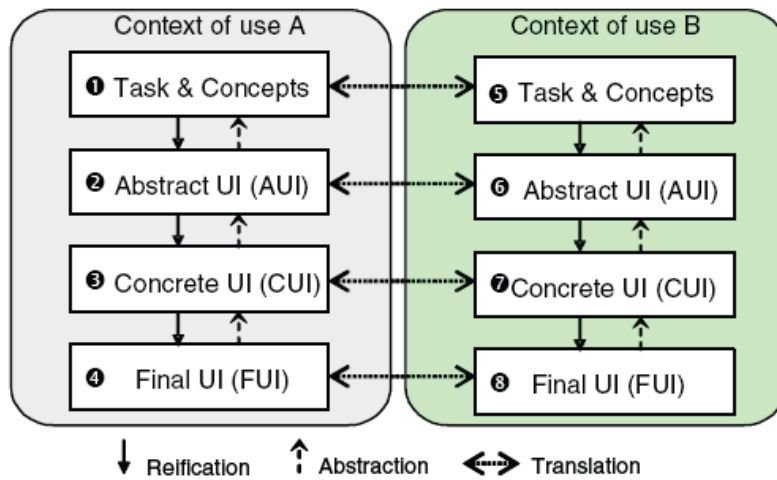



Figure 16 : Simplified reference framework of CAMELEON project [Calvary 2002].

MUI approaches are often pointed out to be complex to set up. The transformations are complex to specify and often hardcoded inside model based tools, leading to unpredictable results [Myers 2000]. However, it is possible to externalize the transformations into a database in order to be able to edit and complete it. One possibility for that is to use an external semantic network that capitalizes the know-how embedded in transformations [Demeure 2006]. This semantic network can then be queried at design time or run time to retrieve transformations that are the best suited to a given context of use.

	<p>Models based approach can cover the whole UI design process. There are models at different level of abstraction, each focus on a particular aspect of the UI (e.g. user tasks, presentation). Among the existing models, the user task and concepts model expresses the semantic of the system from the user point of view. Transformation embeds a know-how of designers.</p>
--	---

Creativity

The notion of creativity relates to different aspects: the creative product, the creative person, and the creative process. In this subsection, I present these three aspects, with emphasis on the last.

Creative Product

The most common definition of the creativity relates to the creative product: an idea is said creative when it is new, original, not common place or routine; and when it is adaptive, functional or effective [Warr 2005] (Figure 17).

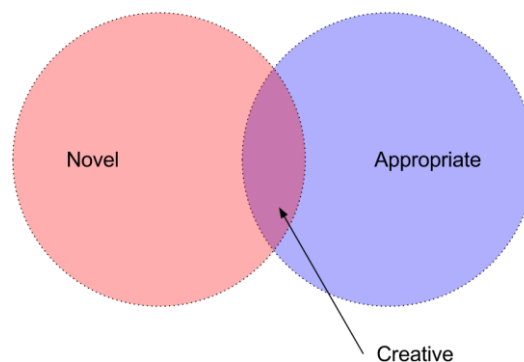


Figure 17: A product is creative if it is both novel and appropriate.

This definition highlights the difference between the notions of novelty and of creativity. While Novelty is necessary, it is the appropriateness of the solution that makes it creative [Warr 2005].

The notion of novelty is not trivial to define. While at first, one would like to say that an idea is novel if it is considered as original, it raises the following question: to whom is it original? Indeed an idea could be new to someone and not to someone else. According to [Boden 1994], novelty falls in two categories: Psychological or Personal novelty (P-novelty) and Historical Novelty (H-novelty). An idea is P-novel when it is new to the designer mind, while it is H-novel when it is new to all minds in the society. In addition to P- and H-novelty, [Suwa 1999] introduces Situated Novelty (S-novelty) which occurs when an idea was not expected in the context in which it emerges. For instance an idea borrowed from another field is S-novel.

In design, the notion of appropriateness relies on the existence of a problem to be solved. As exposed in the previous section, many creative processes start with a problem analysis that identifies the main characteristics of the potential solution. These characteristics, and the evaluation of appropriateness according to them, may vary between and within a domain. In UI design, these characteristics may be aesthetic criteria, usability constraints, functional requirements and so on.

Creative Person

Another dominant approach to creativity is to consider that creativity take its root in particular people, called creative persons. Researches that focused on this aspect have tried to identified the set of traits that are characteristics of such persons. However one should not fall in the layman's but false idea that creative persons are peculiarly gifted with a certain quality that ordinary people do not have [Guilford 1950]. Instead, the general psychological point of view is that all individuals possess abilities to creativity to some extent. Merely, creative people have more abilities than the others. For instance there is a substantial correlation between IQ and creativity, reflected in the common use of the term "genius" (originally coined for creative persons) for IQ high-scorer. Although various creativity tests have been devised to assess the traits of creative individual, their value have been questioned. [Amabile 1983] argues that it is inappropriate to label the results of these test scores as some direct indications of person's creativity.

[Amabile 1983] pointed out three essentials factor to creativity: domain-relevant skills, creative-relevant skills and task motivation. Domain-relevant skill refers to knowledge about the domain, along with technical skills and domain "talent" (e.g. painting requires a certain amount of precision, while skyrocket design requires physics knowledge); Creative-relevant skills relate to the way the individual explore the design space. At last, task motivations correspond to the individual enthusiasm toward the task.

In the development of tools to help designer to design UI interface, one is not only interested in Creative product (*i.e.* creative User Interface) but also in processes that result in the generation of creative products or ideas. Such process are referred as Creative process.

Creative Process

[Shneiderman 2007] identified three schools of thought regarding creative processes. Structuralists believe that creativity comes from the application of methods. They encourage the development of systematic tools. Inspirationalists think that exploring unfamiliar structures develops creativity. They argue for using recreational exploration of unrelated topics. They support the use of image libraries, the practice of sketching interfaces, and the use of concept mapping tools. At last, situationalists recognize creativity as a social process and advocate for collaborative tools.

Design might be seen as a decision making activity on a set of design variable. For example, in the design of a frame, the design variables could be the frame's width and height (left part of Figure 18).

[Gero 1996] proposes that creative design processes, based on the addition and deletion of design variables, have the potential to aid in the design of creative artifacts. For instance, the addition of the radius variable (right part of Figure 18) enables the design of rounded frame that might be seen as creative. However, if the radius variable is set to 0, the resulted frame exhibits no differences from previous frame. Hence there is no guarantee that results of such processes would be considered as creative.

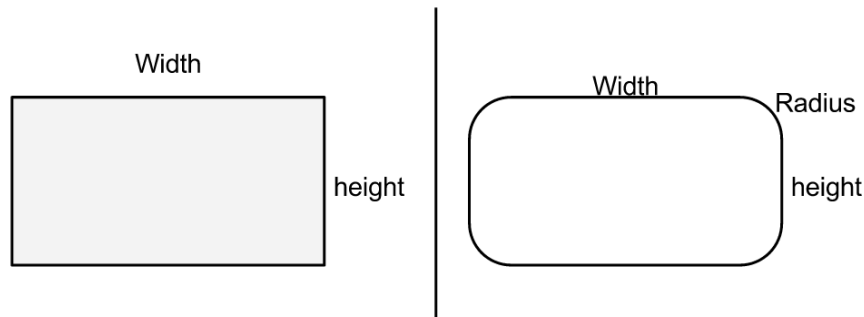



Figure 18: A creative process can lead to the addition of design variable (here: a radius) to an existing problem, resulting in more creative product.

[Campbell 1960] and [Simonton 1999] argue that creativity could be seen as a Darwinian process. In a first step, ideas are submitted to blind variations and recombination, resulting in the generation of a multitude of variants. Variations are blind in the sense there is no way to decide a priori which variations will produce good results on the long term. These variants are then subjected to a selection mechanism. At last, a retention process is applied to the selected variations: selected ideas are preserved in the creator knowledge, and propagated through communication with other creators. Hence, the evolution is reduced to a trial-and-error process. This point of view is close to the one defended by [Buxton 2007] for whom creativity could take place in 1) the meaningful distinct options from which to choose to solve your problem and 2) the definition of the criteria, or heuristics, according to which you make the choice to keep or reject these options.

	<p>I draw several conclusions from these three aspects of creativity. Creative products are both novel and appropriate. They result from an iterative and evolutionary process of combination and variation of ideas, applied by person with sufficient domain skills and motivation. Creative process implies the possibility to add or remove variable from the problem statement.</p>
---	--

Creative Evolutionary Systems

Creative evolutionary systems have been defined by [Bentley 2002], as evolutionary system that address creative problem or help human in creative activity. In this section, we will first study creative evolutionary system in general before studying them when applied to UI design.

Creative evolutionary systems in general

The most common evolutionary systems are Genetic Algorithms (GA). GA is a search technique used to find exact or approximate solutions based on Darwinian evolution principles. They have been formalized in 1975 by Holland [Holland 1975]. GA deal with a population of candidate solutions for a given problem. Each solution is encoded within a sequence of genes. Genes can be sequences of bit,

list of parameters, trees, or complex structure. The important point to keep in mind is that GA manipulates those finite representations, also called genotypes, and not the actual solutions themselves, also called phenotypes. The phenotype of a solution is used uniquely to determine the solution fitness to the problem. The fitness function which discriminates good from bad solution is key to any GA. Once the encoding of the solution and the fitness function established, the GA will evolve solutions through an iterative process. This process is made of four main steps. First, an initial population is generated, *i.e.* a set of genotypes. The population can either be randomly generated or be constituted from previous knowledge about the solution. Here, the key point is that the search is instantiated from multiple points in the problem space, not a single location. Then, iteratively, the phenotype of each candidate solution gets a score that evaluates its fitness with regard to the problem to solve (step 2). Then, some solutions are selected (step 3) and reused to generate a new population (step 4). The generation is done by applying operations on solutions (*e.g.*, mutation or recombination). At last, if no satisfying solutions have been found the process cycles to step 2.

[Sims 1991] applied genetic algorithms to artistic creation and replaced the fitness function with human evaluation. This approach is referred to as IGA [Takagi 2001]. The main drawback of IGA is the user's fatigue they induce for selecting the best individuals over generations. Usually, users can't go beyond 20 generations of 16 individuals [Takagi 2001].

Creative Evolutionary have been largely used in design problem, [Takagi 2001] counts 38 papers that apply IGA to design problems. [Cho 2002] takes a look to Fashion design and evolve dresses. To address the problem of appropriateness, he had recourse to an external database of appropriate dress parts (sleeve, neckline, and skirt).

In 2008, IGA applied to the evolution of neuronal networks have provided outstanding results for image creation as demonstrated with Picbreeder [Secretan 2008] (Figure 19) and in content generation as exemplified with Galactic Arms Race [Hasting 2009]. The success of Picbreeder relies on two mainstays. First, it produces recognizable shape: through evolution, individual go from barely generating circles to producing complex shapes (Figure 19). Second, Picbreeder user can store the individuals in a shared database. Stored individuals can later be used again as starting point for any other users. It is an efficient means to tackle user fatigue and to share individuals among a community.

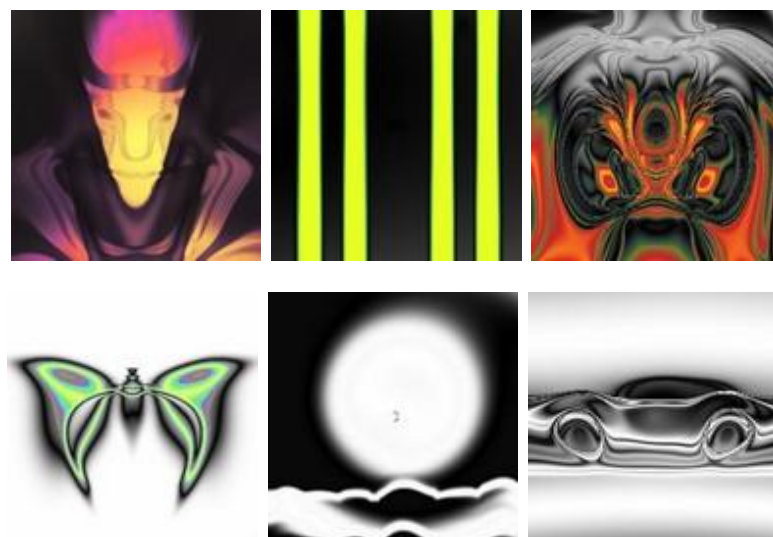



Figure 19: Some images produce by Picbreeder [Secretan 2008].

Both Picbreeder and Galactic Arms Race rely on NeuroEvolution of Augmenting-Topologies (NEAT) [Stanley 2002]. In NEAT, GA are applied to artificial neuronal networks (ANN) --a computational model of biological neuronal networks found in brains -- to evolve both the parameters of the neuronal network and its structure. The particularly good results of NEAT come from its complexification

property. Indeed, in conventional ANN algorithms, each ANN has a fixed topology, *i.e.* a fixed number of node and connection, which limits the search space it can addresses. On the contrary, NEAT approach relies on the possibility to make evolve the topology of neural networks. Hence the search space increases during the process. Neat can address problem of unknown dimensional size, like creative problem where solution require the introduction of new variables [Gero 1996].

	<p>IGA allow to explore the design space and therefore to find new ideas. Innovation can be improved by using complexification through the IGA process. Appropriateness can be improved by using a database capitalizing the know-how.</p>
---	--

Creative evolutionary systems for UI design

In HCI, Monmarché seems to be the first that tried to generate HTML web pages using IGA. In his approach, individuals represent either Cascading Style Sheets (CSS) [Monmarché 1999] or CSS plus webpage layout [Oliver 2002]. In his tools, named Imagine, the population is composed of 12 individuals, each of them is used to generate a web page. The set of web pages produced by the 12 individuals is scaled to fit on a single screen so that the user of the system can grasp them all in one sight. Individual can be partially edited (colors customization) by designers for saving time. At each IGA iteration, designers select best candidates among the 12 web page propositions (Figure 20).

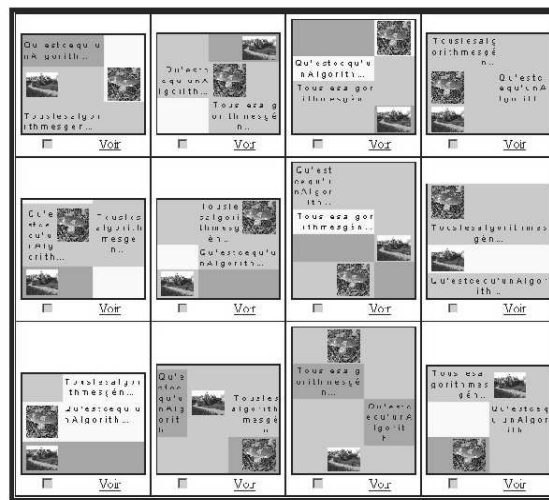


Figure 20: Main part of the user interface of Imagine [Oliver 2002].

Although these works provide interesting results (Figure 21), they also suffer from four main limitations. First, only a subset of CSS attributes is taken into account. It is for instance impossible to change the border of a HTML element. Second, the CSS that are evolved contain a set of predefined selectors (titles of level 1, 2, 3, paragraphs, introduction texts, comments) on which styles apply. Therefore, it is impossible to generate other group of HTML elements in order to apply a particular style (*e.g.* grouping an image and a text related to this image to add a border around them). This limitation is probably due to the lack of semantic presents in a web page. Indeed, the HTML document only provides a syntactic structure and there is no standard way to represent semantic information in it. Hence, there is no guarantee that the layout evolution is not going to separate semantically close HTML elements or, on the contrary, to visually group semantically distant HTML elements (see for instance Figure 21). This lack of semantic knowledge leads to a third limitation of Monmarché's

approach: the impossibility to substitute HTML element by other ones. Such a possibility could be of great interest for adapting a web page. One could for example imagine to substitute a text entry representing a date by a calendar or any equivalent widget. At last, there is no enrichment of the individuals. Both the set of selectors and the set of style attribute are fixed along the evolution process. As a consequence, no new elements can appear (e.g. a label in front of a text entry).

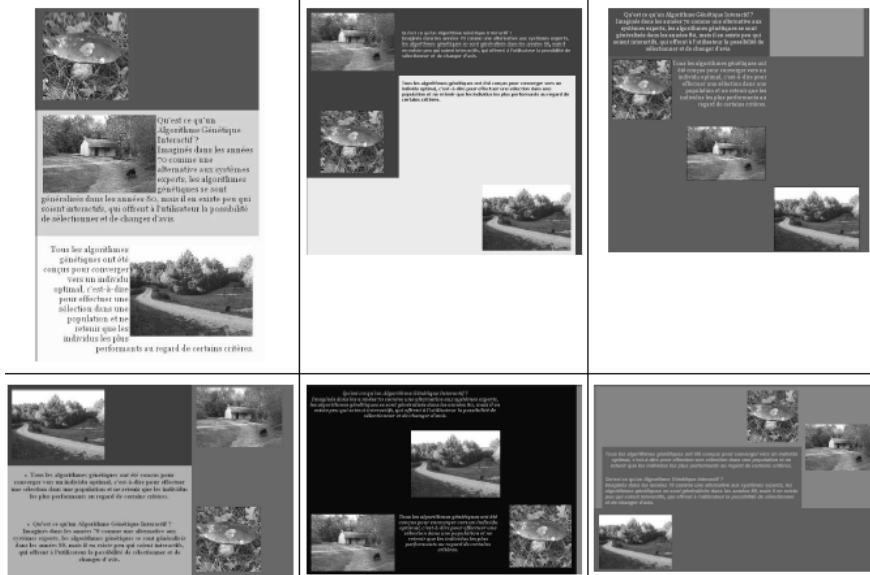


Figure 21: Some results that can be obtained with Imagine [Oliver 2002]. Both layout and CSS style have been evolved.

[Quiroz 2007] explores IGA to generate XUL UIs (Figure 22). He uses wider populations (hundreds of individuals) but presents a subset of the population only to designers. The designers have to select the best and the worst individuals. Then, an interpolation/extrapolation algorithm is applied to the rest of the population to automatically evaluate individuals. The evaluation of the individuals is partially done by the users inputs, which is referred to as “subjective evaluation”, and partially done by automatic evaluation, which is referred to as “objective evaluation”. In [Quiroz 2007], the objective evaluation concerns the 1) contrast between background and foreground colors and 2) the alignment of widgets, done by construction as widgets were placed on a two columns grid.

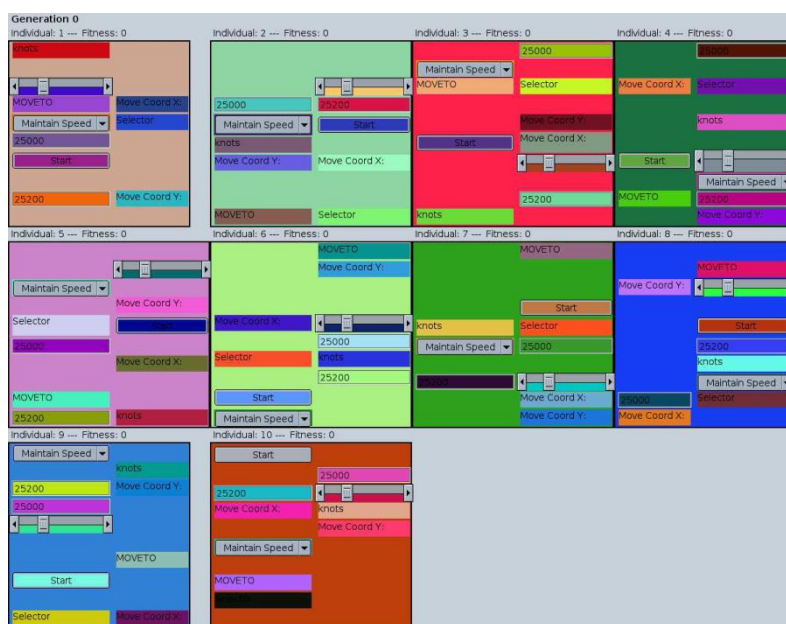


Figure 22: The population of XUL UI from which the designers have to select the best and the worst.

Although at a first glance this approach seems to be promising with regard to the user fatigue, it raises a fundamental question that is not answered by the author: can a UI be automatically compared with “bad” and “good” samples? In [Quiroz 2007], the evaluation is based on one basic criterion only: the UI blueness. As automatic evaluation is not solved so far, I prefer not to base my proposition on a so strong hypothesis. In addition, the layout is simplistic in [Quiroz 2007] and it is not possible to replace widgets with semantically equivalent ones. At last, there is no complexification of individuals along the IGA process.

[Quiroz 2009] also explores the use of collaborative IGA to foster the creativity. The interesting aspect of this work relies on how the collaboration between designers is done: Each designer is working alone, but when one designer evaluates an individual to be satisfactory, it is exported to the other designers (Figure 23). By doing so, [Quiroz 2009] find out that the UIs created collaboratively were ranked slightly higher with respect to the creative criteria. However, it has to be noticed that even collaboratively created UIs obtain relatively low scores.

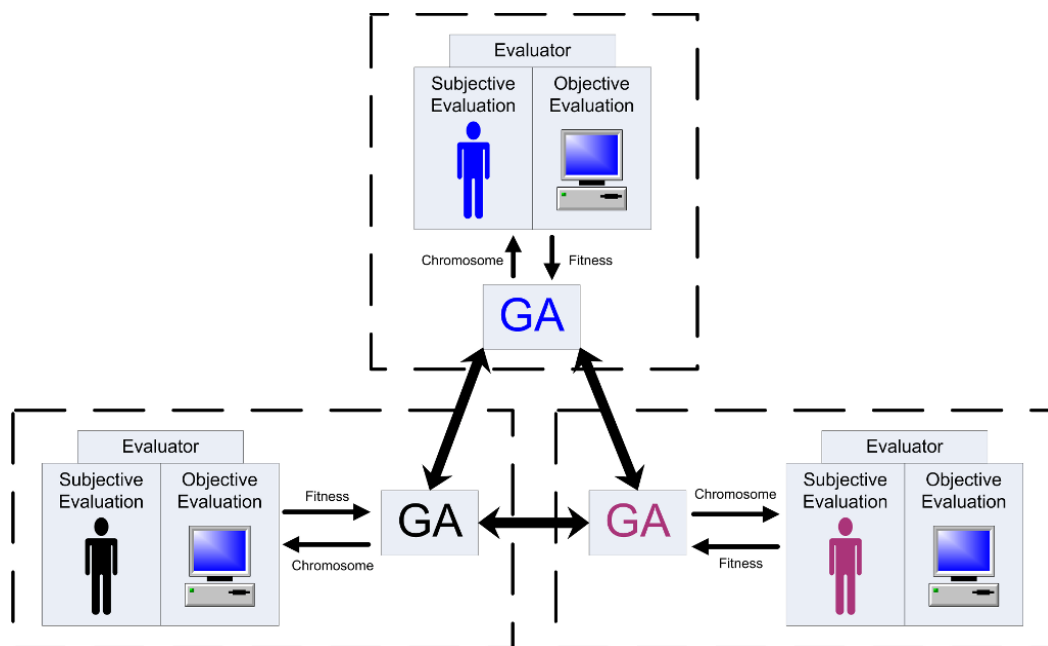


Figure 23: Collaborative IGA in [Quiroz 2009]. The best individuals of each designer are shared with the others designers.

Last, [Plessis 2008] proposed to enhance the work done by [Quiroz 2007] by focusing on the layout problem. Instead of aligning widgets in a grid, [Plessis 2008] uses a tree representation of the widgets (Figure 24) where parent nodes are layouts (e.g. grid, flow). The tree structure and the type of layout node are then mutated along the IGA process to produce new UI. Results are visually better than the ones obtained by [Quiroz 2007] but suffer from similar drawbacks: there is no semantic description of the UI that could enable widget substitution and there is no complexification along the IGA process.

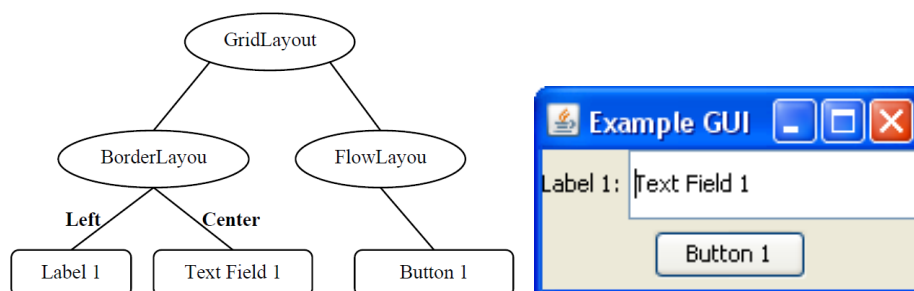



Figure 24: Left) Example of a tree used in [Plessis 2008], parent nodes are layouts. Right) The corresponding user interface.

One interesting aspect of [Plessis 2008] is about the evaluation: unlike Monmarché (who provides only informal evaluation by showing obtained results) and Quiroz (who is mainly interested in reducing user fatigue and evaluating produced UI), Plessis asked evaluators of his system (three HCI experts) to complete a short questionnaire to gather qualitative data. The answers pointed out the problem of the semantic (they expected related fields, such as “name” and “surname”, to be grouped), the lack of control over the evolution process (“you cannot control the mutations”) and in particular the impossibility to fix some elements while making others evolve.

	<p>State of the art of IGA applied to UI design suffer from several lacks:</p> <ul style="list-style-type: none">• no semantic representation of the UI to design is available• no complexification of the UI is possible• the elements to be evolved inside the UI are predefined and cannot change• There should be a way to specify that some part of the UI are fixed and that some other can evolve
---	---

Conclusion of the state of the art

UI design is an iterative process. First iterations are dedicated to the exploration of numerous design options through sketches. This exploration is crucial: the more numerous designs are considered, the more chances there is to find good designs early in the design and therefore save time and money. Next iterations produce more and more precise UI designs, from low fidelity prototypes to the final design. Despite the fact that, on the overall, the number of considered designs shrink down along the design, new designs are proposed and existing ones are removed all along the process. Making explicit and keeping trace of the design rationale implied in the choice of discarding or adding designs is important for designers to remember, or learn, the reasons that led to the final design.

MUI approaches are good candidates to represent the designs along the design process. Indeed, every UI design can be seen as a model and relationships between different designs or design rationales can be expressed by transformations. One UI model hold my attention; the concept and tasks model (C&T). The C&T model aims at representing the semantic of the UI from the user point of view. It describes what tasks can be done by the users while interacting with the system. At last, it is independent of how the UI will be concretely designed. I consider that each C&T model defines a design space of concrete UI compatible with it. Enabling the designer to easily explore this design space is the goal of this work.

Another interesting aspect of MUI approaches is the possibility to capitalize parts of the know-how of designers (e.g. how to represent a task to the users) in a database. Exploiting this knowledge should enable a more relevant exploration of the design space.

IGA are good candidates for helping designers to semi-automatically explore the design space. Previous works based on IGA applied to UI design suffer from several limitations, in particular the lack of a semantic model of the UI and the impossibility to complexify candidate solutions. [Quiroz 2007] and [Plessis 2008] apply their IGA on individuals that represent a final UI (e.g. a XUL UI). With such approaches, representing the semantic of UI elements is not obvious: some elements have a functional semantic related to the user tasks and concept (e.g. a text entry), some others just structure the UI (e.g. a layout), or ensure the user guidance (e.g. a label), or pleasure. In my opinion, taking all these cases into account would be something too complex to be effective.

A different approach is taken by [Monmarché 1999]. He applies IGA to individuals that represent Cascading Style Sheets (CSS). These CSS are applied to an HTML document to produce the final

web page. I consider this approach to be more interesting as it provides a clear separation between the document structure (with HTML) and the graphical style (with CSS). However, the design space generated by CSS is fairly limited to color, font, border and, to some extent, layout specification. It is for instance impossible to change the HTML structure with CSS. The other limitation of the approach comes from the HTML document itself, which is subject to the same criticisms than the ones done in the previous paragraph for XUL interfaces.

I believe that IGA, coupled with MBUI approaches and coupled with a database of know-how, can be of a great interest in HCI for facilitating the design space exploration and thus inspiring designers to produce creative UI designs.

3. Contribution

I start by presenting a case study that will support my point. Then I develop one by one the steps of the process. Finally I present Magellan [Masson 2010] a prototype of my approach.

Running Case study

I chose as a case study a simple Instant Messenger Client (IMC) for both clarifying my point and the experiments I conducted. An IMC is a software that enables a real-time text-based communication between two people. This IMC has the following functionalities: The user has a profile which includes its name, photo and availability status (e.g. Online, Busy, Idle and so on); the application manages a list of contacts, in which the user can start a conversation with, add, or remove a contact. In the experiment version, the contact consists of only a name. A conversation is text-based (i.e. I do not consider audio or video conversation) and between the user and one contact (i.e. I do not consider group conversation). It consists of a list of messages already sent or received, and a text area to specify a new message. Figure 25 presents a COMET task model of the IMC. Each node represents a COMET which can be a task or a task operator. Inside each node, the first line describes the type of the node (e.g. an interleaving), the second line is the name of the task or task operator. Finally, the last line describes into brackets the concepts that are used. For legibility reason, not all concepts are present.

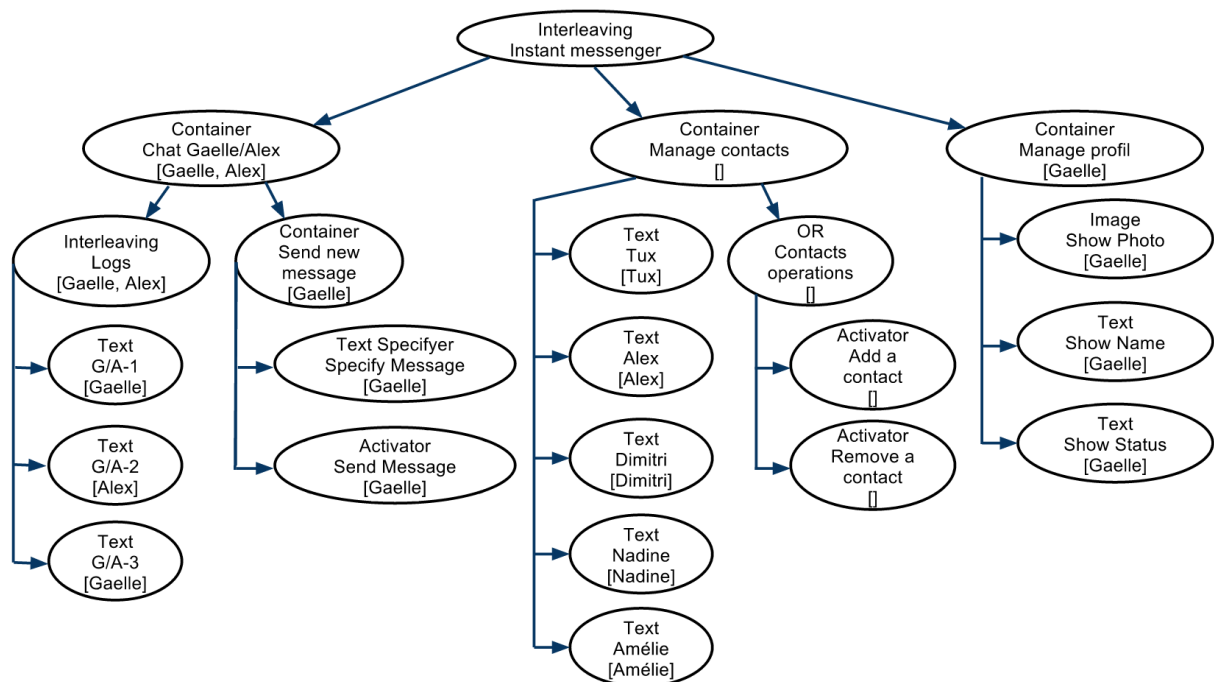


Figure 25: COMETs have been used to build the concept and task model of the instant messenger client.

Overview of the approach

My approach is based on both IGA and MBUI. The fusion of IGA and MBUI operates on two levels: First, we evolve transformations and second, genetic operations of the IGA are conditioned by a) the problem, represented by the task and concept model describing the UI to design and b) Expert knowledge on UI design capitalized in a database of models and transformations that express a form of know-how of HCI designers (Figure 26). To represent this database, I chose to use the semantic network presented in [Demeure 2006]. This semantic capitalizes how tasks and task operators can be represented by CUI. It enables queries such as “Find a presentation for task T that is close to the P

presentation”. Such queries are useful to find appropriate widget variations for a task. For instance, the semantic database can inform that “*Choice of a date* tasks are represented with text entries” can be mutated into “*Choice of a date* tasks are represented with calendar”

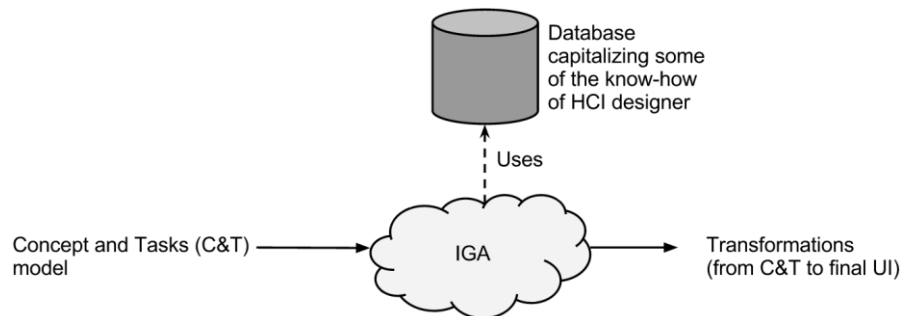


Figure 26: IGA apply on a concept and task model to produce transformations. A database capitalizing parts of the know-how of designers helps to build relevant transformations.

The deployment of GA starts with choosing a genetic representation for the solution candidates. Then, it consists of several steps: initialization of the population, application of genetic operations (such as mutation and inbreeding), and evaluation and selection of the individuals. The following subsections describe the choices I have made for each of these steps.

Genetic Representation of Individuals

I encode my individuals as a set of transformations that produce concrete UI (CUI) from a C&T model. I claim that evolving transformations is more powerful than directly evolving a CUI. A higher level of expressiveness can be achieved by manipulating transformations than direct manipulation of the CUI. Indeed, most of the interface is already fixed in a CUI (e.g. basic layout, widgets or workspace have already been defined) and GA operates only to tweak some parameters of the interfaces. On the contrary, transformations enable the construction of the interfaces through all level of abstraction (*i.e.* Definition of the workspaces, widget and layout selection, etc.). By using transformations, I introduce a greater level of variations while ensuring that the resulting UI designs are conform to the original task.

The transformations used in this work are simple rule-based transformations. A transformation is described as a couple <Selector, Translator>. The selector identifies one or several elements from the C&T models (e.g. select “text specifier” tasks that are child of an interleaving task operator). The Translator expresses how to translate identified tasks and task operators into concrete UI elements (use a list of radio buttons to represent choices).

One major aspect of my work lies in the fact that, unlike [Monmarché 1999, Quiroz 2007, Plessis 2008], neither the features nor the elements to be evolved are coded in the problem definition. The features possibly evolvable are capitalized in a semantic graph (the GDD, see [Demeure 2006]). New features added to the GDD are automatically taken into account during the evolution. While [Quiroz 2007 and Plessis 2008] evolves independently each widget, and while [Monmarché 1999] groups UI elements before the IGA process starts, my approach enables groups to be constituted all along the IGA process. As a consequence, designers can also get inspired about what grouping method is interesting or not.

I argue that the evolution of the selectors is in line with [Gero 1996] model of creativity. It introduces new variables that might not have been imagined by the designer during the problem definition. For example, a transformation that initially applied to all the buttons in the interface might evolve in a transformation that applied to all the 'OK'-button.

Individuals encoding

There is an infinity of manners to encode transformations. I have implemented two of them: list and tree. In the first version of my prototype, each individual was represented as a list of transformations. Unfortunately, this format displayed poor performance. The reason was that groups of selectors evolved independently from each others, resulting in many overlapping. To circumvent this problem, the latest version of the prototype is based on a tree representation of the transformations (Figure 27). Each node represents a translator and each edge represents a part of a selector. The complete selector corresponding to a given translator is composed of the concatenation of the selectors from the root node to this translator node. Each node contains a list of couple attribute-values that represents how the selected tasks and task operators have to be translated into CUI elements.

Figure 27 illustrates a transformation tree; all transformation trees start with a root node that targets the top level task of the C&T model provided as input (in our case, the Instant Messenger Interleaving of Figure 25). From this node, two transformations follow: on the left, the selector targets all the containers that descend from the root node, each of them is translated into a frame; On the right, all the interleaving that descend from the root node are selected and translated into tabbed panels with a black background and a white foreground color. The last transformation is applied from these interleavings, it select their children that convey the concept of “Gaelle” concept and change their background into white and their foreground color into white.

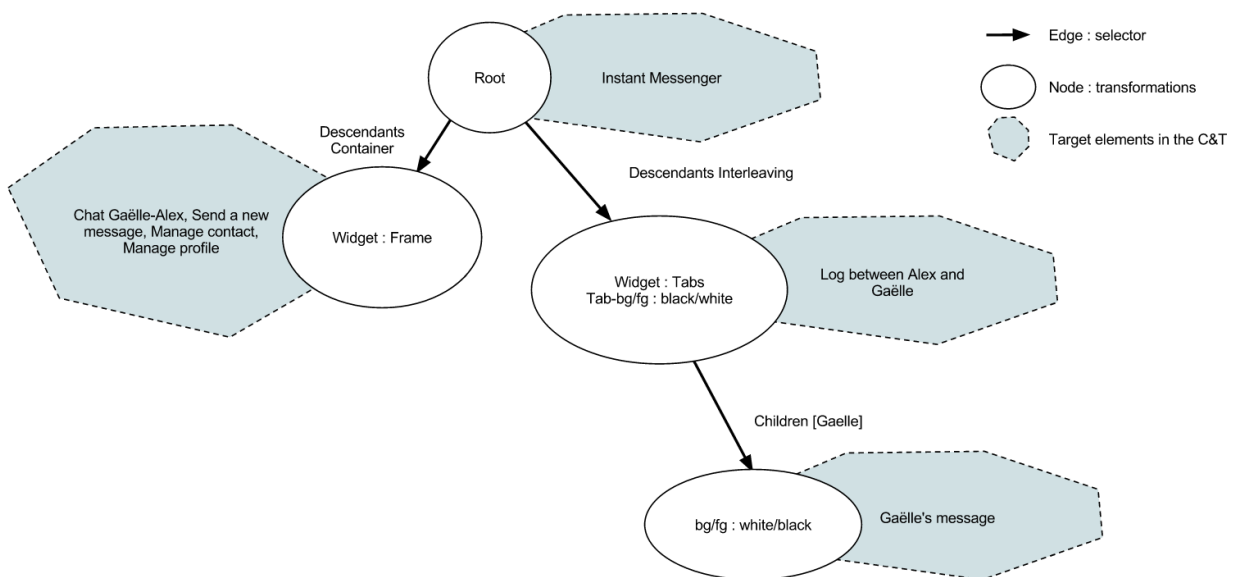


Figure 27: Example of a transformation encoded as a tree.

The tree representation offers the possibility to easily create new transformations on the basis of existing ones. For instance, Figure 28 describes the transformation after a new transformation that targets the container marked with the concept of has been added below the left node of Figure 27.

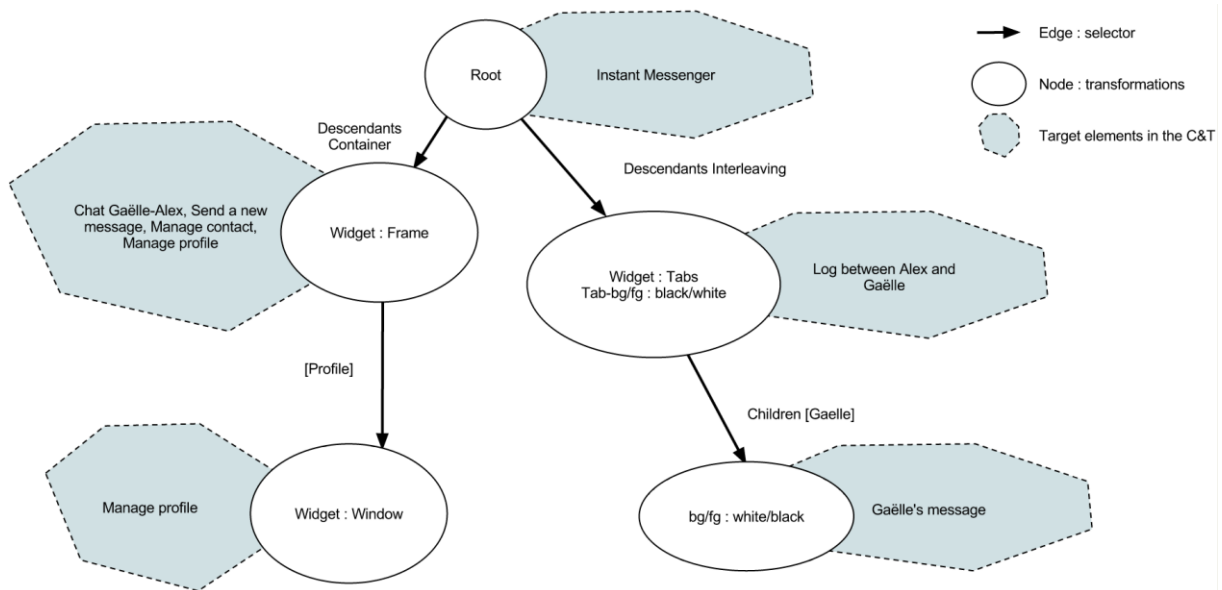


Figure 28: A new transformation can be added by plugging a new node to the tree of Figure 27.

The next sections describes in details the different steps of my IGA, starting with the initialization of the first population.

Initialization

Initialization has to deal with two concerns: the size of the population and the generation mechanism. The size of a population is crucial: if too small, there is a risk to converge toward “bad” or undesired solutions; if too large, evaluation of solution becomes time consuming. Picbreeder [Secretan 2008] shows that it is possible to obtain good results with IGA dealing with a population of 16 individuals. In my approach, the default size of the population is 16 but it can be parameterized by the user.

I envision several initialization mechanisms. I briefly describe them below and indicate whether I have implemented them or not:

- Void initialization (implemented): In his approach to augment topology, [Stanley 2002] argues that evolving individual from scratch leads to finding minimal solutions. However, in our case, starting with minimal individuals would cause the first UI designs to be poor. Hence, the first user evaluations would be boring and pointless.
- Random initialization (implemented): Starting with already evolved individuals resolve this problem. Random generation is done by applying successive mutations on a void individual.
- Manual initialization (implemented): With manual specification, designers can either manually produce transformations or import them from previous uses of the IGA. This approach is useful for supporting scenario such as importing external transformations from other tools.
- Examples based initialization (not yet implemented): Designers could guide the system by providing a set of UI design or UI sketches. The system would be in charge of finding transformations producing visually close UI designs to the ones provided by the designers.

Mutation

Mutations are one of the two core genetic operations that can be applied on individuals. Tree structure mutations add or remove nodes and their related edges. By doing so, it is possible to complexify the individuals and thus the transformations to be applied to C&T. I detail how mutations are done on nodes and edges.

Node mutations

Node mutations are from far the more extended and complex mutations that I have implemented. They can modify the task presentation of the targeted elements in several ways: change the widget, change specific features (color, size etc) of the widget, or encapsulate the generated widget into a more complex CUI graph (e.g. adding labels to a text entry).

All the possible mutations are encoded in the semantic network database presented in Figure 26. I extended this network by specifying, for each interactor, the list of features (e.g. color, height, order of the elements in a container, etc.) that could be submitted to evolution. Moreover, I added in the network functions describing how to manipulate each feature. In particular, each feature is related to functions enabling to get a new value (e.g. get a new color). It has to be noticed that these features can be shared among a lot of interactors (e.g. the background color feature) or can be very specific to one widget (e.g. the foreground color of the selected tabbed pan). With this approach, the design space to be explore for a given C&T model can be extended simply by enriching the database with new features.

A default widget is associated to each task type (see Figure 29-left). Widget mutation occurs only if all the targeted elements belong to the same task type. Thanks to the database, appropriate widget can be substituted to the default one. All specific features of the widget are also added to the translation expressed in the node, thus enabling the parameterization of the widget (see Figure 29-right).

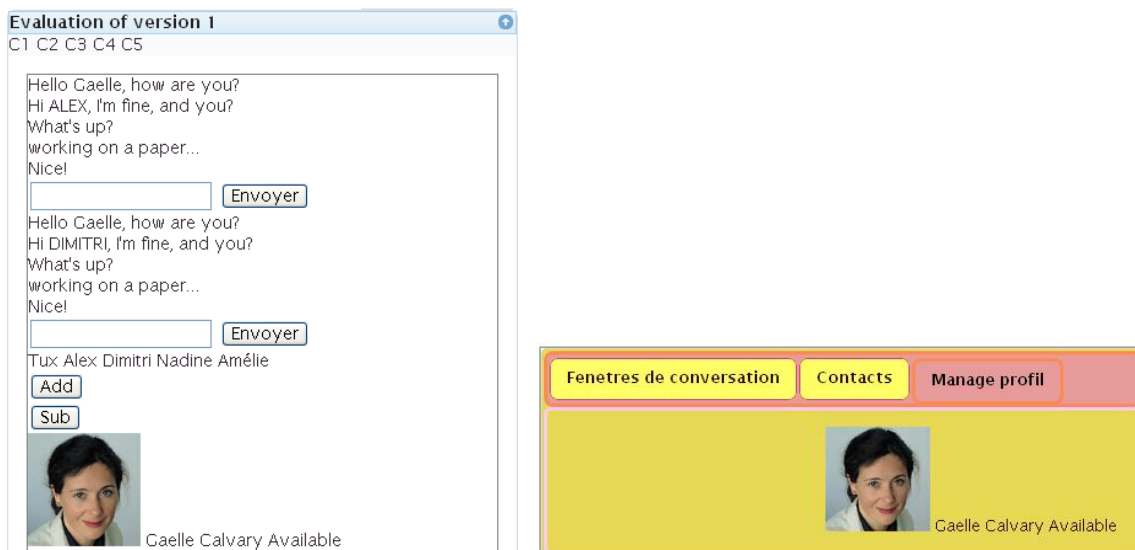


Figure 29: Left) A void individual produce an UI with default widgets. Right) By substituting widgets to default one and setting related features, the resulting UI change dramatically.

Background and foreground colors evolution is particular and requires extended clarification. It follows two principles: one aesthetic and one ergonomic. The aesthetic principle comes from my previous experiences with UI generation using IGA. Indeed, IGA tend to produce UI with a too many colors (one per element) that do not go well together. To remedy to this problem, [Quiroz 2007]'s fitness function favor UI with low contrast between widgets. I argue that the low contrast between widget is wrong since it penalizes UI based on two opposite colors, such as black and white. I have envisioned the use of a color theme for each Individual.

A color theme is a set of predefined colors. Every color in the UI is based on one color of the theme or on a small variation of one color (such as lighter or darker, to add rollover effects or gradients). A color theme can be either generated **randomly**, using **color theories**, or **specified by users**. Random generation of a color theme reduce the number of color present in the UI, but does not resolve the disparity between them. Color theories explain how to choose colors that should match well together

(Monochromatic, Complementary, Tryadic, tetrahedric, etc.) but there is no guarantee that the set of produced color would be interesting. At last, users can specified their own color theme. Various websites (colourlovers.com, kuler.adobe.com) offer the user to create, share and rate color palettes. Hence, these website capitalize knowledge about interesting color themes. I selected the latest option for my prototype: color themes are represented by a palette of 5 colors retrieved from the kuler.adobe.com website.

In plus to the aesthetic principle, I applied an ergonomic principle to choose colors. I enforce readability of the text by ensuring a minimum contrast between the foreground and the background color (W3C : <http://www.w3.org/TR/AERT#color-contrast>). For any widget the background and foreground color specification is done as follow :

- Select a background color, which is either: inherited from its parent, or a color in the color theme;
- Optionally apply a small variation on the color (such as enlighten or darken the color);
- Form a list composed of all the color of the color theme plus all their possible variations (lighter, darker and so on) that are contrasted enough with the selected background;
- Select a foreground color in this list.

The last kind of node mutation is called “widget enrichment”. Basically, it consists in adding to the widget chosen to represent a task (or task operator) other widgets (e.g. images or labels). Enrichments can results in better **guidance**. For instance, in Figure 30-left, messages sent by Gaëlle are enriched by its photo (taken from her profile). Enrichments can also result in **shortcuts**. For instance, in Figure 30-right, each contact of the list is merged with a presentation of the add and remove commands.



Figure 30: Enrichments' mutation can result in guidance (on left, photo are added to messages) or shortcuts (on the right, each contact is merged with add and sub commands).

Edge mutation

Edges mutations modify selectors. The modification could lead to a broader selection (e.g. “select children” being generalized into “select all descendants”), a more narrowed one (e.g. “select all descendants” being restricted into “select children”) or a completely different one (e.g. “select children” becoming “select descendant containers”). Edge mutations resulting in empty selection are discarded, as well as mutation resulting in equivalent selectors. These two rules ensure that every node and edge is useful for the transformation.

I use three types of selectors: children, descendant and refinement. Refinement can be coupled with children or descendant. It consists in selecting a subset of currently selected C&T element with respect to their type (e.g. interleaving, choice) and/or their associated concept (e.g. Gaëlle). The bottom-left edge of Figure 28 illustrates a refinement done with respect to the “profile” concept.

Inbreeding

Inbreeding is the other core genetic operation that can be applied on individuals. There are several ways to achieve inbreeding between two (or even more) individuals. So far, I have implemented a mechanism which randomly selects branches of each parent to create a new individual. If two nodes are related to equivalent selectors, only one of them will appear in the offspring. However descendants of the sidelined node, are reconnected the one that have been selected. Figure 31 highlight such crossing. Another option that I want to explore in the future is to use human guidance to achieve inbreeding. The idea is to promote branches from each parent identified as “interesting” by designers.

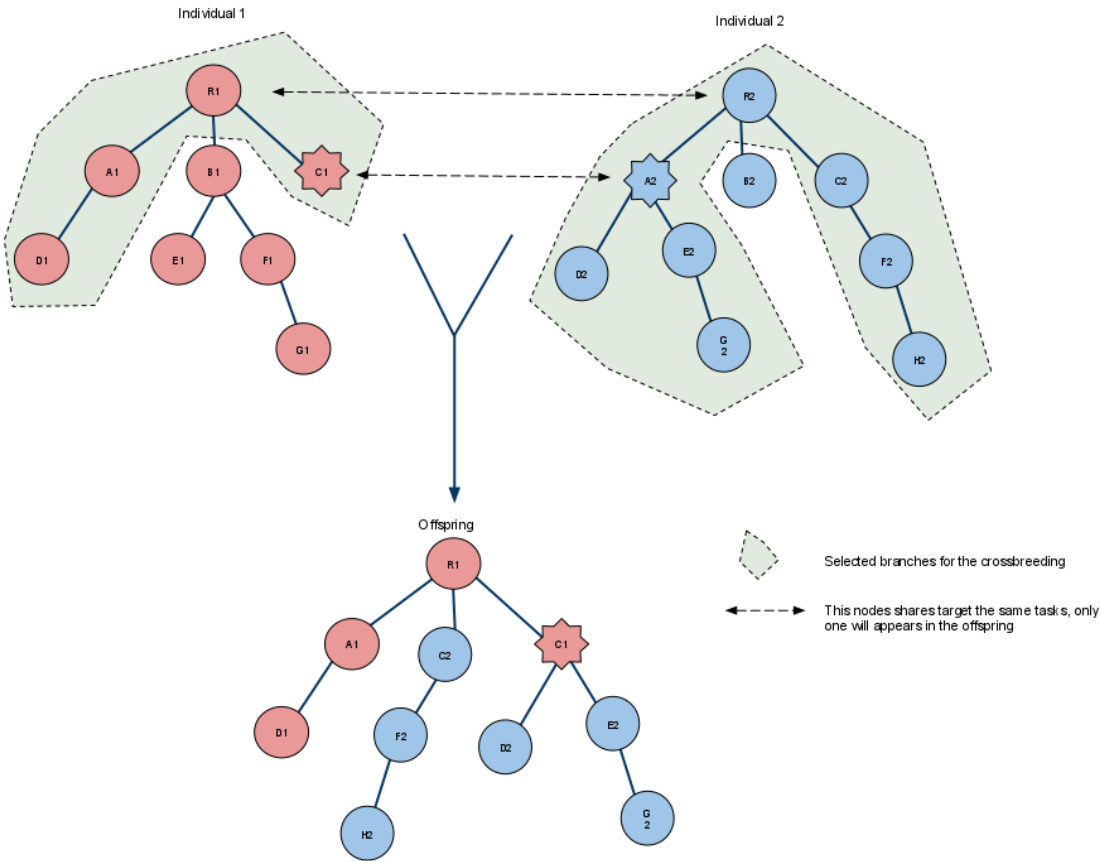


Figure 31: Example of crossing. Similar nodes are merge in a unique one.

Fitness Evaluation

The fitness evaluation aims at assessing the quality of each individual with respect to a given problem. In creative design, the quality correspond to both novelty and appropriateness of the solution. In IGA the user evaluates the quality of each individual according to objective or subjective criteria [Quiroz 2007].

I did not implemented evaluation based on objective criteria in my prototype. Rather, as seen in previous sections, I implemented genetic operations that ensure by construction the respect of ergonomic criteria based on colors. I did this choice for two reasons: first, implementing an automatic

evaluation is something complex and second, presenting to the designers UI design that we know to be inappropriate uselessly increases the cost of the evaluation.

I decided that the designer should be the only ruler on board, *i.e.* the evaluation is entirely performed by the designer. There are several ways to score candidates. The simplest scale is a binary notation, as done in Picbreeder [Secretan 2008]. An extension of this notation is to attribute scores on a wider range (*e.g.* 1 to 10, or reals between 0 and 1). These two solutions are modeled with evaluation components in my prototype.

One evaluation method that I consider as very promising is the partial evaluation. The designers would identify the specific parts of the UI that they find interesting and the evolution would be based on that information. This method would offer possibilities to increase the potential of the inbreeding as well as the mutations. This improvement will be part of future version of my prototype.

Selection of the individuals

Once the user has evaluated the individuals, the question is: which elements should I keep for the next iteration? Several selection methods exist in the literature. One can cite rank selection (best candidates are selected) or roulette wheel selection (selection probability proportional to the score). Elitist strategies may be applied. They consist in keeping the best individuals unchanged through generations in order to avoid regression. Resurrection enables the designer to recover an ancestor candidate and re-inject it into the current generation. The approach I have taken is to let the maximum of flexibility to the user. The next section explains how the user can combine the different elements presented to build its own IGA.

Flexibility of the process

For flexibility reasons, Magellan is based on a modular, fully customizable flowchart architecture. I make it possible for designers to build their own IGA by assembling a series of components. Each component implements one of the operation presented above (*i.e.* evaluation, mutation, selection, etc.). Connections between components convey population of individuals.

I propose a toolbox of five flowchart component types: the four classical operations in IGA (see for example the flowchart of Figure 32): initialization of a population (green), mutation (pink), crossing (red), and evaluation of individuals (yellow). In addition, I add flowchart controls components (blue) such as “union” or “filter”. By using all these components, designers are able to build a wide variety of IGA (*e.g.* Figure 32 illustrates a basic convergent/divergent IGA).

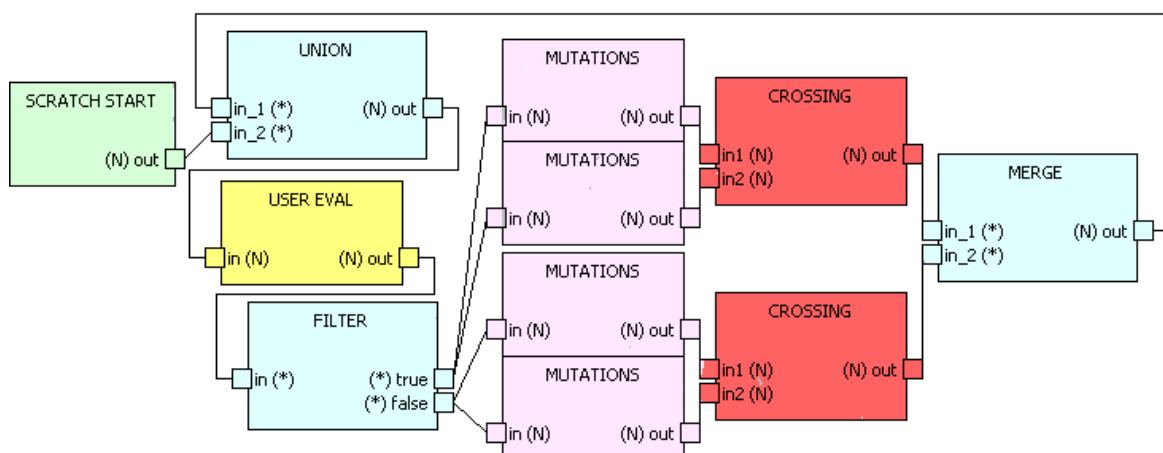


Figure 32: Flowchart of Convergent/Divergent IGA. Highly rated individuals are submitted to a low mutation rate, and cross together. Poorly rated individual are submitted to a high mutation rate and cross together.

Prototype

I have implemented Magellan [Masson 2010], a prototype to test the validity of the approach I proposed. I first describe the technological choices I have made for the realization of this prototype before describing the user interface of Magellan.

Magellan has been built upon the COMET toolkit [Demeure 2008]. COMETs are polymorphic task level interactors covering all levels from C&T to final UI. They are assembled into graphs. A graph of COMET is equivalent to a concept and task model. Each COMET contains several presentations that can be dynamically added or removed. The presentations contained in each COMET can use different rendering technologies (e.g. HTML, TK, OpenGL). The COMET architecture ensures that presentations of assembled COMETs are technologically compatible (e.g. HTML presentations of COMET C1 are plugged with HTML presentation of COMET C2).

As presented in [Demeure 2006], I use a semantic network to implement the database that stores the possible presentations of a given task. For now, this network contains few widgets for each task. It has been extended to include the features that are evolved in Magellan.

Finally the COMET toolkit comes with CSS++, a transformation language. A CSS++ transformation sheet is composed of a set of rules. Each rule is composed of two elements: a selector and a translator. The selector identifies the COMETs to be translated (e.g., all the top level interleavings). The translator specifies the way the selected COMETs are translated (e.g. set the background to white, display interactors as tabbed panel). In CSS++, the choice of the interactor for representing a given task can rely on the semantic network. In Magellan, the transformations encoded in the individual are translated into CSS++ transformation prior to the rendering.

My approach is not dependent of this technological choice. Another possibility would have been to use CTT for the C&T model along with XSLT for the transformations. However, I am not aware of any knowledge database capitalizing XSLT transformation models applicable to CTT models.

Magellan UI

The UI of Magellan is divided in two pieces: A workflow editor (based on the TK rendering technology of COMETs) dedicated to designers who want to tune their own IGA and a web based UI (based on the HTML/AJAX rendering technology of COMETs) dedicated to the exploration of the design space.

Figure 33 shows the workflow editor. It is composed of four parts: the top right part is a canvas dedicated to edit the workflow. Components can be placed, removed and connected to specify the workflow. The active components are highlighted. The left part of the UI contains a list of components that can be drag and dropped on the canvas. The five top buttons (Reset, Play/Pause and Step, Save and Load) control the flow and enable to save and load flowcharts. The bottom right panel presents the list of individuals that are currently evaluated by the designers. Finally, the bottom left part displays the current C&T COMET graph on which transformations are applied. The user can select an individual in the right list and a task in the graph to highlight on the web UI the related UI elements (see Figure 34).

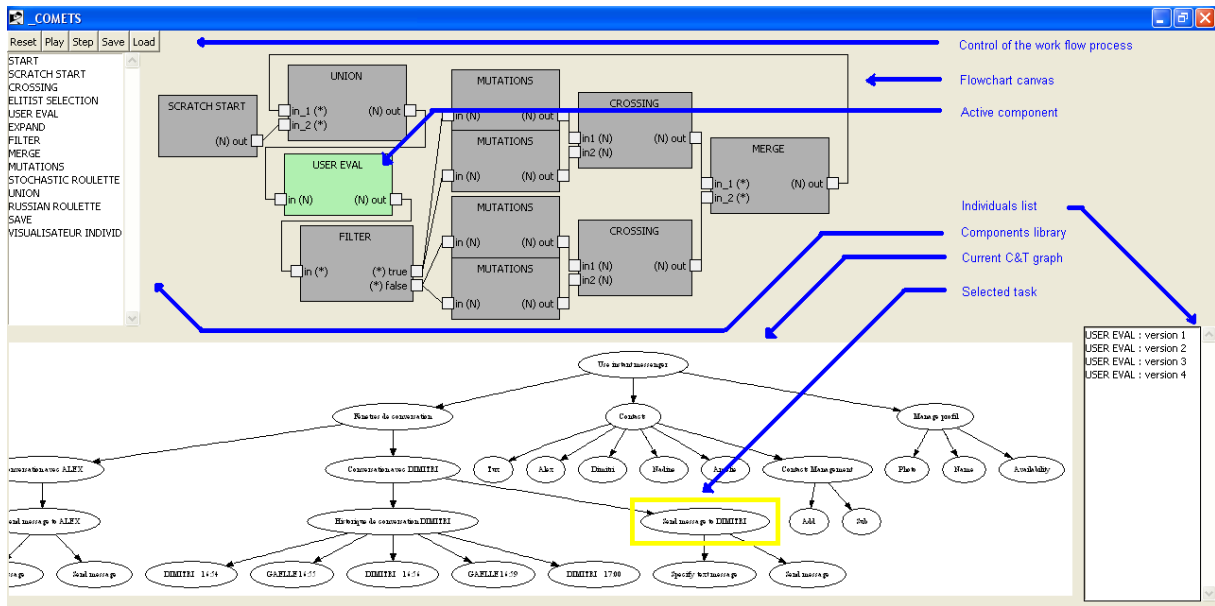


Figure 33: Magellan's Workflow editor. The list on the left contains evolution components. The graph on the middle is the current evolution workflow. The highlighted component is the active one.

Figure 34 shows the web based UI of the user evaluator component. The five top right buttons are the HTML presentations of the “Reset”, “Play/pause”, “Step”, “Save” and “Load” COMETs also present in Figure 33. UI designs are displayed into frames. For each of them, the color theme is displayed. Designers can select and deselect UI designs by clicking on them. When selected, a UI design is surrounded by a thick green border. The “Validate scores and continue” button enables to validate designers’ choice. From the workflow point of view, it propagates scored UI design to the next components.

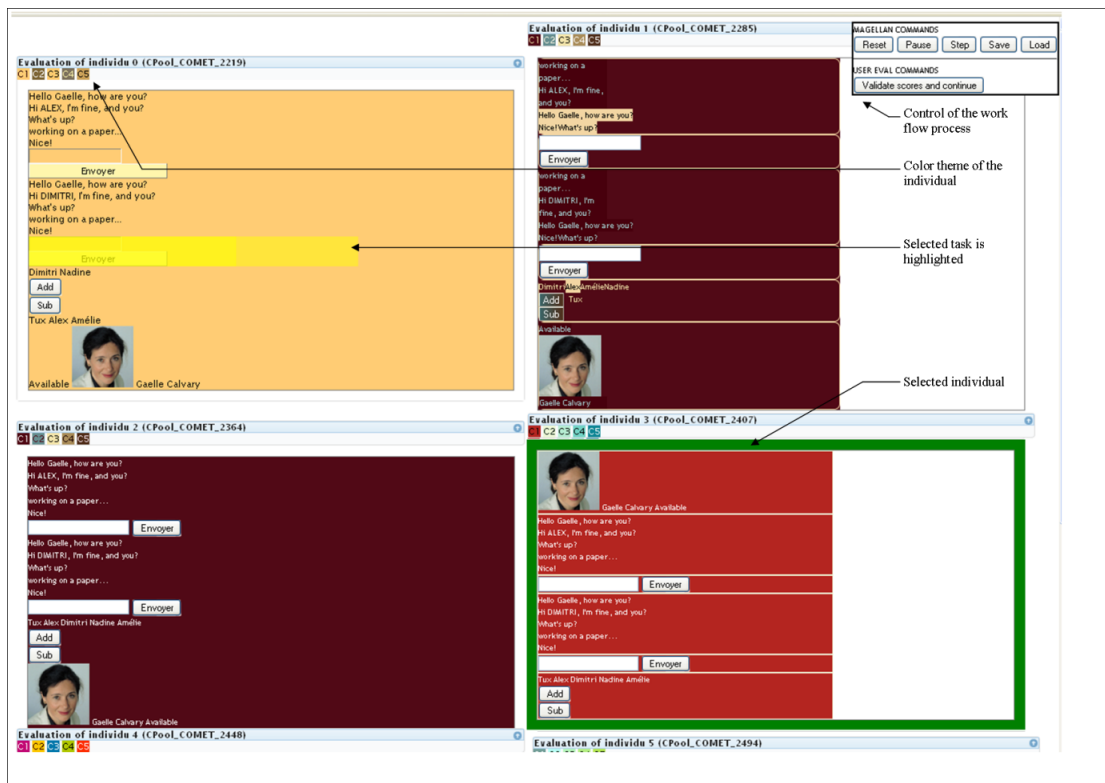


Figure 34: The web based user interface of Magellan. Users can click on thumbnails to select the ones they prefer.

4. Evaluation of the approach

In previous works using IGA in creative support tools for UI design, researchers focused on assessing either the diminution of user fatigue [Quiroz 2007] or the greater value of the product [Quiroz 2008, Cho 2002]. Only [Plessis 2008] put in place a qualitative evaluation, based on a questionnaire, to gather data about the design process as supported by their tool.

We think that such a qualitative evaluation must be done. Indeed, despite the promising benefits of using IGA for UI generation, the works done by [Monmarché 1999] and [Quiroz 2007] had little impact on the HCI community. We think that the problem probably comes from a misunderstanding about how IGA can be used in the design process and what kinds of functionalities are expected from designer in an IGA based tool.

The main hypothesis tested in this evaluation is the usefulness of using IGA to produce examples to foster creativity of designer during early stage of UI design. We also want to know **how** IGA can be used to be useful. In particular, I believe this approach will prove particularly well with beginner designers or people with little inspiration. In this evaluation, my prototype Magellan is used as a general prototype for the IGA approach in UI design since it is functionally analog to previous work such as Imagine [Montmarché 1999] or IGAP [Quiroz 2009] (except the collaborative aspect) . I apply the semi-directed interview method to build my experimentation. This method is detailed here after.

Semi-directed interviews

In order to test the adequacy of creative support tools and to observe the diversity in behaviors, I faced HCI designers to collect a maximum number of track of possible developments. To complete this information collection, the recommend method in both sociology [Combessie 2007] and HCI [Dey 2006, Hindus 2001] is the semi-directed interview. This method belongs to the qualitative methods, such as participative observation, focus-group [Bastien 2002, Bruseberg 2001] and the wizard of Oz. Semi-directed interviews enable the emergence of a maximum of ideas, opinions or habits, even in the case they are infrequent in the population under study. The underlying objective of semi-directed interviews is not a quantitative evaluation of these behaviors or needs, but to list as many. In sociology it is recommend to achieved at least 20 interviews, to reach the saturation point of this list. Indeed in practice, one can observe behaviour redundancy and a dry in new and original ideas after about 20 interviews. To obtain a large panel of ideas and increase the variability, participants recruitment must be focused on potentially motivated people whom sociodemographic characteristics differ from each other. For creative support tools for HCI design, I will focus on people interested either by HCI or design. To increase variability HCI people will come from student to teacher, while designers will come from various domains (e.g. product designer, graphic designer).

Semi-directed interviews are done in face-to-face dialogue following an interview grid. This grid includes the set of themes that must be debated during the interview. The first theme is relatively large to slowly induce the participant into the interview. For instance first questions are related to his habits about informatics, advantages and shortcomings of software he uses and so on. Following themes focus on the object of the study.

Experimental protocol

To test the adequacy of IGA as creative support tools, I chose to use the semi-directed interviews method along with Magellan. Indeed, Magellan is a prototype not only of my work but also of the general use of IGA as creative evolutionary system. I ran a first experiment of 11 interviews with both HCI people and designers with various profiles in term of genre, age, and profession. I merely conducted half the recommended number of interviews, due to time constraints.

The system that we used for the evaluation consist of a DELL precision M6300 (core2 duo 2.4 Ghz and 2.0 Go of RAM) laptop running with Windows XP sp3. The TCL interpreter used for the COMET [Demeure 2008] runtime was “Active TCL 8.5.7”. The internet server was EasyPHP and the internet browser used to render the produced UI was Firefox 3.6.

Magellan was set as follow: The population size was 16 individuals, the input C&T was the simple Instant Messenger is given in Figure 25. Figure 35 represents the workflow scheme used during the experiment (colors were added to help the visualization). The process starts with the green components (left of Figure 35): **SCRATCHSTART** generates a population of 16 individuals, all void. The **MUTATIONS** blocks initialize the individuals with 7 random transformations. The **UNION** block concatenates the individuals received in inputs. The yellow components are responsible for the user evaluation; the **USER EVAL** component is related to a user interface that enables designers to evaluate UI designs produced by the individuals received in input. The **FILTER** split the individuals received in input. Individuals scored better than 0.5 are outputted on the “true” output while the others are outputted on the “false” output. In pink, the **EXPAND** components duplicate, shuffled and repeated the received individuals until there are 16 of them. Then, the **CROSSING** component inbreeds the individuals received as input “in1” with the ones received as input “in2”. The resulting inbred individuals are outputted and merged with individuals produced along the cyan path (bottom of Figure 35). The cyan path aims at generating 16 new individuals. It starts with a trick; the first **FILTER** component ensures that no inputted individuals will be transmitted as output; The **EXPAND** component behave like the **SCRATCHSTART** component if the input is. The cyan **MUTATIONS** acts like the green one. At last, the **MERGE** blue component merges the individuals received in inputs. It was set to keep 8 elements from the pink path (originally from the user selection) and 8 from the cyan path (the new individuals). Finally, a small mutation pressure is applied to all the individuals before a new cycle starts.

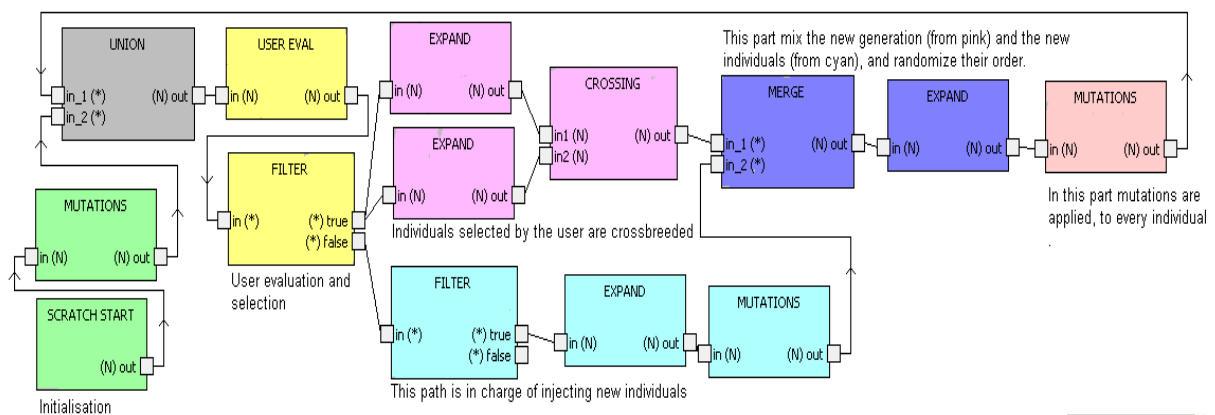


Figure 35: Magellan's workflow used for the experiments.

The experimental protocol (See appendix B) has been designed with Nadine Mandran. Nadine is an expert engineer in experimental methods from Marvelig¹. Each session concerned one participant at the time. First 5 sessions were conducted by Nadine Mandran, Alexandre Demeure and I, the next four by Alexandre and I. Finally the last one was done by me only. Each interview consists of three parts: First, the participants filled a form about their HCI design practice and their relations with creativity. In the second task, we presented the instant messenger software to the participants and asked them to propose several sketches of UI design. The participants were provided with papers and color pens and had about 45 minutes to come up with their design. After a small debriefing on this

¹ Marvelig, is a platform for scientific experimentation, which aims to develop collaborative works between teams of the LIG.

task, we presented Magellan to the participants. They were asked to think aloud while using the prototype. Finally, we debriefed the participants about the prototype; in particular we asked them what were the advantages and the shortcomings of the approach, and what they would like to improve in such a tool. At the end of the session, we presented Picbreeder [Secretan 2008] as another creative support tool to feed the discussion.

Participants to the experiences include 8 males and 3 females. Table 1 describes the participants, they can be classified in four groups regarding their relation to HCI:

- HCI experts came from the research field of HCI, include 1 HCI assistant professor (E2) and 3 Ph.D students (E0, E3 E4). E4 has the particularity to have more than 20 years of experience in industry.
- Programmers are expert code developers. Include 1 webmaster (E8) and 1 senior programmer (E5).
- Designers include 1 product designer (E1) and 1 graphic designer (E10).
- Beginners have little knowledge about programming or HCI. 3 Students were recruited from a Master 2, they have followed at least one HCI course (E6, E7, E9).

Table 1: Participants of the experimentation.

ID	Relation to HCI	Description
E0	Programmer	Ph.D student, web design
E1	Designer	Product Designer
E2	HCI expert	Assistant professor
E3	HCI expert	Ph.d student
E4	HCI expert	Ph.D student + 20 years of experience in industry
E5	Programmer	Senior Programmer
E6	Beginner	HCI student
E7	Beginner	HCI student
E8	Programmer	Webmaster
E9	Beginner	HCI student
E10	Designer	Graphic Designer

Interview synthesis

I sum up the results of the experiments in several sections, each focusing on a specific topic discussed with the participants.

Creative Process

Most of the participants follow an iterative and incremental process as expected. They first started by an analysis of the problem, and picked out one or two main flaws of current approaches. For instance, (E0), (E2), (E3) and (E4) considered the number of windows involved in many IM to be too important

and proposed solutions to tackle the problem. In general, participants proposed only one design track, and stopped before the end of the given time for the session. Only (E1) proposed several tracks of ideas, and was willing to take more time to propose more ideas (see appendix C). (E1) is one of the two professional designer interviewed, he was really motivated by the experiment, unlike the other designer who produced one or two sketches. [Amabile 1983] framework could explain those results: In order to be creative, a designer should have Domain skills, Creative Skills and Motivation. Professional Designers are likely to have improved their creative skill through their formation and career. Therefore, they are more likely to produce several tracks. (E1) have little experience in web design and was highly motivated, while (E10), on the contrary, lack of experience and motivation. (E8) had extended domain skills and motivation for the task, but did not consider herself as a creative person. Students' motivation seemed to have been tainted by apprehension of disapproval although we clearly stated that we would not judge the quality of the production. At last only designers reported to use tools such as Photoshop to refine their ideas before going to production. Others participants reported to directly implement their ideas.

Even though the recruitment was voluntary not focused on professional HCI designer to gather as much information as possible. The interviewees that displayed Domain-relevant skills, Creative-relevant skills and motivation were definitively the designers.

Collaboration

“Ca avance toujours plus quand on est plusieurs” -- E1

Currently, Magellan does not support collaborative work. Therefore, the following statements are not issued from comments of user but rather come from answers to questions about their design habits, and collaborative work they have done.

As stated by [greene 2002] and [shneiderman 2007], collaboration is essential to creative activity. (E1) reported that designers work in team, with experts from the different fields concerned by the project: programmer, graphic designer and final user. This team work permits to establish a dialog about the design and enables the designers to step back from the design. I identify from these interviews two kinds of collaborative works; within experts (other designers, programmers, domain specific experts) and with final users of the application. Creative support tools should support collaboration of experts during the design. However, it should be considered that any experts may have a different work approach; as a consequence, collaborative support should enable experts to bring efficient solutions to domain-specific problems (e.g. ergonomic criteria, aesthetic aspect, interaction, etc.) (E7). Regarding the dialog with the final user, tools like Magellan can be used to present a myriad of prototypes to probe the user needs (E1).

Constraints

“Le designer est un entonnoir” -- E1

Designers are subjected to numerous external constraints while designing. Through the interviews, I identified four kinds of peoples that are at the origin of the constraints that guide the designer in their tasks: Experts, Clients, Peers and final user.

Expert constraints come from various domains. They can be part of the know-how of the designer, or come from domain-specific experts. (E1) stated that he saw the designer as a funnel. The role of this “funnel” is to gather and assemble domain-specific expertise to produce creative design with a minimum knowledge about these domains. Reported expert constraints are related to usability and ergonomic criteria (e.g. (E1) and (E2) reported the semantic of the colors in HCI), and technological

constraints (e.g. (E6) reported that one of its first step was to select a technology so to determine early what is possible or not).

Client constraints include functional specification along with graphical constraints. They are likely to evolve during the project. Surprisingly designers seem to be under the pressure of their peer and the final user, resulting in implicit fashion constraints. (E0) reported that web designer tends to follow trends both in terms of graphics, technology and functionality. (E2) stated “Il y a des canons à respecter” (there is some standards to meet), referring not only to standard guidelines, but also to evolving tendency in the design. It appears that a right design is thus a moving target: what is good now might not be good later on. This emphasizes the importance of the designers’ full control in the selection process of IGA; they are the only ones that could have a taste of the current tendency.

One can wonder what influences have other designer’s works. And in general what the designers’ inspirations are.

Inspiration

The interviewees usually explore existing designs before or while designing. I distinguish two types of explorations:

- 1) The first exploration type is about reviewing similar websites or applications (E1, E7, E8, E9) or specialized websites in design (E0, E6, E8). The objective of this exploration is to collect knowledge, tendencies and to identify problems and solution. Tendencies researches are not futile, but results from the following consideration: if the design is too unusual, the user will possibly reject it.
- 2) The second exploration type is about random exploration. They usually occurs in the form of non targeted searches (e.g. during the everyday life, for another project) (E1, E3 E4). Such searches are not restricted to domain specific UI (e.g. medical website in the case of a medical project) nor specially related to UI design. For instance, inspiration can be drawn from analogy with other design: E1 proposed a design for the contact list of the Instant messenger with large icons, he explained that the origin of his idea came from a phone dial: “Des gros boutons comme un cadran de téléphone” (big buttons like the ones used on a telephone) (E1).

Professional designer (E1, E10) also have recourse to picture library they have build across previous research. Inspiration from other designs is not limited to the look and feel. Borrowed elements could also be architectural (E4, E2). As for programmers, it is frequent that code snippets are extracted from other designs; Look and feel is then modified to suit their project.

Innovative elements

Innovative elements cited by participants during the preliminary questions have been grouped into three categories: Aesthetics Elements, Look and feel, and functional core.

Participants seemed to be principally interested by aesthetic elements while browsing for examples (e.g. images, background, logo and other graphic elements). This surprisingly contrasts with the sketching part of the experiment (where participants were asked to produce sketches): only one participant (E4) used meaningful colors in his sketches. Some others used colors to distinguish their comments or to highlight some part of their sketches. While Magellan does not propose examples with images, it makes an important uses of colors, and I expected positive reactions. Even if colored UI were better received than black and white UI, it appears that UI were rarely chosen for their colors. Rather, UI were rejected if the color displeased to the participant (for example pink themed UIs were largely disliked).

Two reasons could explain these results:

- 1) In the second step of the experiment, participants were asked to quickly sketch UI, thus they used a visual language proper to early sketching [Buxton 2007]. The visual language used by Magellan is closer to high-fidelity prototype than sketches; Hence, it could explain the fact that produced sketches used no color, while black and white UI proposed in Magellan looked like in-achieved regarding colored UI.
- 2) Participants were still looking for early designs while using Magellan, they were more interested in the general organization of the interface than in “details” such as the background color. A good color theme is not sufficient to make a good design, but a bad color theme is sufficient to make a bad one.

Participants made several propositions about how using colors in Magellan: 1) They wanted to have the possibility to select a color theme for all the proposed UI (E1 and E10, among others, explained that the color theme is often imposed by the client). 2) They wanted to have the possibility to try different color themes for a specific UI (E8, E7) and 3) to have the possibility to directly change a specific color of the theme (E2).

Even though layout was considered as an innovative element by only 5 participants when asked in the first part of the experiment, all of them started selecting or commenting the UI presented by Magellan with respect to their layout. According to (E1), the work on the layout aims at saving the real estate of the screen or discriminate concepts. The layout is innovative when it fulfills this constraint and breaks away from standards. A lot of criticisms were about the column layout, which is the default html rendering: This layout produced UI which were difficult to read. Critics were also made about layouts that produced UI with insufficient distinction between the tasks. These designs were largely rejected while the ones with clear separation and guidance were preferred. Finally, several participants had questions about the interaction with the produced UI: “What happened if I open a new conversation?”. Even if Magellan does not address that problem, those questions help participants to envision shortcoming about some designs (e.g. opening conversation aside contacts cause problems when several conversations have to be opened).

From this review, I consider two possible developments for Magellan:

- 1) There should be a way to select the levels of abstraction and refinement of the produced UIs. For instance the designer could first evolve abstract user interfaces (*i.e.* only workspaces), focusing on the layout, then move onto more concrete levels to specify the details of the UI (*i.e.* interactors to be used, colors theme, etc.).
- 2) The examples provided should be fully functional. For instance, it should be possible add or remove elements in a list (e.g. the list of contacts in our case study).

Magellan

In general, interviewees found Magellan “not perfect but interesting” (E1). Most of them had the feeling that Magellan had proposed UI that looked like what they had proposed. The main flaw exhibited by Magellan comes from the injection of new elements at each iteration of the IGA. Users were confused and wondered why some of the proposition did not correspond to their previous selection. However, the injection of new individuals was not considered intrinsically bad. Rather, the lack of visual identification of the new elements was the cause of participant’s trouble.

Participants were sometimes reluctant to select an a UI design because a part of it displeased them. Conversely, they expressed that they would like evolve the whole interface but a part of it. This support our idea to develop partial selection and evolution of the UI designs,

Among the participants interviewed, the one that were identified as designers thought that creativity should come from themselves (E1, E10). (E10) expressed that Magellan appears to be a limit to its creativity since its design coverage was limited (few interactors possible for each task, not all layout

were possible, etc.). As (E10) came has a graphical background, we presented him Picbreeder to feed the discussion and check whether his reluctance to use Magellan came from the domain (UI design) or the approach itself (IGA). It turned out that he was reluctant to the approach, making the same types of critics to Picbreeder. Our hypothesis is that for some people, especially the more creative ones, creative design tools should be more centered on the manual edition of designs, with peripheral automatic exploration of variants of designs made by the human.

Every participant felt the need to directly move, resize or change the color of some elements of the UI. As a consequence, we plan to add the possibility to handle manual edition in next versions of Magellan. In particular, participants proposed:

- 1) to have the possibility to direct edit some UI designs. The modifications done should be present in the next generation produced by the IGA.
- 2) to have the possibility to inject sketches directly in the UI designs proposed by Magellan. This could be useful to add new widgets or to delimit layouts.
- 3) to have a central edition area (E3, E9) where they could pick up elements from the proposed UI designs to build their own design. The concept of a webpage editor where the user can borrow elements to construct its own webpage have been proposed recently by [Lee et al. 2010]. However they used examples manually extracted from the web.
- 4) to have the possibility to fix some parts of the designs. For instance, (E1) wanted to fix the color theme, while (E2) expressed the need to fix a particular layout and the use of particular widgets.
- 5) to have the possibility to select parts of the UI designs (instead of selecting the whole UI design).
- 6) to have the possibility to identify “good” UI but also to identify “bad” or “neutral” ones.
- 7) To have the possibility to edit the task model and to explicit relationships between it and the proposed UI designs. Indeed, by the end of the experiment, we presented the mechanism behind the screen to the participants. At last two users (E8 E9) were interested in the task model and wanted to see on what tasks corresponded to what parts on proposed UI designs.

Conclusion of the evaluation

This first experiment went very well and reached the goals I had set. It confirmed the overall benefits of the approach. Most of the participant found interesting features in Magellan. Some of them, like the professional designer, would consider using this tool. The main criticism done by participants was more about a peculiar configuration of the IGA process (the injection of new individual at each cycle) than about the approach in general.

However, it appeared that the functionalities offer by Magellan were not sufficient. I expected that providing a semantic model and enabling the complexification of UI design would fulfill the designers need. It turned out that this was not the case. The opinions collected point that designers feel a strong need to edit the proposed design. They also feel the need to evaluate parts of the proposed UI designs. These needs are not fulfill by current approaches [MonMarché 1999, Quiroz 2007, Plessis 2008, Masson 2010]. This could explain why they had so small incidence.

These interviews confirm the improvement that I had planned (manual edition, partial selection) and make me realize the importance of layout in early design. These remarks will be taken into account in the future version of Magellan.

Finally, this small study on a dozens of people had offered me the possibility to become familiar with semi-directed interview (recruitment, conducting and analysis). It will be my base for a more extensive investigation after modification of the prototype.

5. Conclusion and future works

My master thesis work is about enhancing existing approaches for supporting creativity in User Interface (UI) design. I have identified in the state of the art that UI design is an iterative and evolutionary process. Each iteration of this process is composed of expansion steps where new design options are explored and reduction steps, where the less promising design options are discarded. As pointed out by the state of the art, creativity is actually present at every steps of the design process.

Creativity has been considered from different perspectives in the state of the art. In my work, I consider the creativity from the perspectives of creative product and creative process. Creative products are not only novel but also appropriate. Translated in terms of UI design, it implies that produced designs must fulfill ergonomics and functional requirements. More than the creative product perspective, I am mainly interested in the perspective of creative process. Indeed, UI design is a creative process. It has been theorized that such processes can be seen from a Darwinian point of view in which ideas recombine and evolve to produce new ideas.

Interactive Genetic Algorithms (IGA) have been explored to support creativity in UI design. Nonetheless, the works based on this approach had little impact on the HCI community. However, I think that these approaches are really promising. I put forward the hypothesis that this lack of success is related to: 1) the absence of model describing the semantic of the UI and 2) the impossibility to enrich UI designs along the process. In order to overcome these limitations, I decided to fusion IGA with model based UI (MBUI) approaches. Indeed, MBUI approaches make it possible to express the semantic of the UI through concept and tasks models. In addition, transformations model can achieve enrichment of UI designs.

To assess my approach, I have implemented Magellan, a prototype that evolves web based UI. My work leads to the publication of a short article of 6 pages at the international conference of rank A, EICS 2010 (see the article joined in appendix A).

I conducted a dozen of semi-directed interviews to get feedbacks about the usefulness of the IGA based approach. These interviews confirmed that IGA are an interesting approach. However, contrary to my hypothesis, the lack of success of IGA is not only due to the absence of semantic and enrichment. It turned out that participants felt a strong need to be active participant in the design process. This includes for example the possibility to edit the proposed UI designs.

The results of these interviews lay the basis of my future works. In the short term, I plan to develop a new version of Magellan taking into account remarks made by participants. In particular, it should include:

- Direct edition facilities: Designers should be able to edit proposed UI as well as to add sketches of new widgets or layouts.
- Partial evaluation: Designers should be able to identify parts of the designs that they found “good” or “bad”
- Partial evolution: Designers should be able to fix parts of the UI.

In the medium term, I plan to conduct an evaluation of the new Magellan prototype. In particular, I want to compare it with the approach proposed by [Lee 2010]. One goal of this evaluation will be to compare whether designers are best inspired by real life example (Lee’s approach) or purposely generated examples.

In the long term, I would like to explore different extensions of my work. First, I would like to explore the generation of UI sketches in place of final UI designs. In particular, I will focus on sketching layouts

as it appeared to be an important concern during the experiments. Secondly, I would like to focus on the aesthetical aspect of the UI designs. To do so, I plan to evolve complex shapes comined with images composition and color themes. Thirdly, design process is a social activity, consequently I plan to explore how to efficiently include collaborativity in my approach. In particular, I will pay particular attention to the different roles involved in the UI design (designers, programmers, end-user, usability experts, etc.). Finally, I am interested in evolving UI interactors and interaction techniques.

6. Bibliographie

- **Amabile 1983:** Amabile, T. The social psychology of creativity: A componential conceptualization. *Journal Of Personality And Social Psychology* Volume: 45 Issue: 2 ISSN: 0022-3514.
- **Bastien 1993:** Bastien, J. M. C., & Scapin, D. L. (1993). *Ergonomic criteria for the evaluation of human-computer*. Technical report, INRIA.
- **Bastien 2002:** Bastien, J. M. C., & Scapin, D. L. (2002). *Les méthodes ergonomiques: de l'analyse à la conception et à l'évaluation*. *Ergonomie et Informatique Avancée (Ergo-IA'2002)* (Biarritz, France, 8 au 10 octobre) (pp. 127-143), Biarritz, I.D.L.S.
- **Berti 2004:** Berti, S and Correani, F. and Mori, G. and Paternò F. and Santoro, C. *TERESA: a transformation-based environment for designing and developing multi-device interfaces*. In proceedings of CHI 2004, pp 793—794.
- **Bentley 2002:** Bentley, P. and Corne, D. *Software risk management: principles and practices*. *Journal IEEE software*, pp 32—41.
- **Bias 1994:** Bias, R. G. and Mayhew, D. J., (Eds.) (1994). *Cost-Justifying usability*. San Francisco: Morgan Kaufmann Publishers.
- **Bodden 1994:** Bodden, M.A. *Dimensions of Creativity*. Book published in 1994. ISBN-13: 978-0-262-02368-9.
- **Boehm 1991:** Boehm, B.W. *Software risk management: principles and practices*. *Journal of IEEE software*, pp 32—41, 1991.
- **Bruseberg 2001:** Bruseberg, A., & McDonagh-Philp, D. *Focus groups to supports the industrial/ product designer: a review based on current literature and designers' feedback*. *Applied Ergonomics*, volume 33, 27-38, 2001.
- **Buxton 2007:** Buxton, B. *Sketching User Experiences: Getting the Design Right and the Right Design*. 448 pages, Morgan Kaufmann (March 30, 2007). ISBN-13: 978-0123740373.
- **Calvary 2002:** Calvary, G. and Coutaz, J. and Thevenin, D. and Limbourg, Q. and Souchon, N. and Bouillon, L. and Florins, M. and Vanderdonckt, J. *Plasticity of User Interfaces : A revised reference framework*. In proceedings of TAMODIA'02, Bucarest, 2002.
- **Campbell 1960:** Campbell D. T. *Blind variation and selective retention in creative thought as in other knowledge processes*. *Psychological Review*, 67, 380-400, 1960.
- **Cho 2002:** Cho, S.B. *Towards creative evolutionary systems with interactive genetic algorithm*. *Journal of Applied Intelligence* volume 16, pp 129—138, 2002.
- **Combessie 2007:** Combessie, J.C. *La méthode en sociologie*. Collection Repères. Edition la découverte. Juillet 2007.
- **Coyette 2004:** Coyette, A. and Faulkner, S. and Kolp, M. and Limbourg, Q., and Vanderdonckt, J. *SketchiXML: towards a multi-agent design tool for sketching user interfaces based on USIXML*. In Proceedings of the 3rd Annual Conference on Task Models and Diagrams (Prague, Czech Republic, November 15 - 16, 2004). TAMODIA '04, vol. 86. ACM, New York, NY, pp 75—82.
- **Demeure 2006:** Demeure, A. and Calvary, G. and Coutaz, J. and Vanderdonckt, J. *The COMETs Inspector: Towards Run Time Plasticity Control Based on a Semantic Network*. TAMODIA'2006. Hasselt, Belgium, 23-24 october, 2006.

- **Demeure 2008:** A. Demeure, G. Calvary, K. Coninx. *COMET(s), a software architecture style and an interactors toolkit for plastic user interfaces*. In Proceedings of DSV-IS 2008.
- **Dey 2006:** Dey, A.K. and Sohn, T. and Streng, S. and Kodama, J. *iCAP: Interactive prototyping of context-aware applications*. In Proceedings of Pervasive 2006.
- **Dragicevic 2004:** Dragicevic, P. and Fekete, J. *Support for input adaptability in the ICON toolkit*. In Proceedings of the 6th international Conference on Multimodal interfaces. ICMI '04. ACM, New York, NY, pp 212–219.
- **Erdmann 1971:** Erdmann, R.L. and Neal, A.S. Laboratory vs. Field Experimentation in Human Factors — An Evaluation of an Experimental Self-Service Airline Ticket Vendor. In Human Factors: The Journal of the Human Factors and Ergonomics Society, Volume 13, Number 6, December 1971.
- **Gajos 2004:** Gajos, K. and Weld, D.S. *SUPPLE: automatically generating user interfaces*. Proceedings of the 9th international conference on Intelligent user interfaces, 2004, p 100.
- **Gajos 2005:** Gajos, K. and Weld, D.S. *Preference elicitation for interface optimization*. In Proceedings of the 18th annual ACM symposium on User interface software and technology. UIST'05, 2005, pp 173–182.
- **Gero 1996:** Gero, J.S. *Creativity, emergence and evolution in design*. Journal Knowledge-Based Systems volume 9, 1996, pp 435–448.
- **Goldschmidt 1991:** Goldschmidt, G. *The dialectics of sketching*. Creativity Research Journal, 4 (2), 123--143, 1991.
- **Greene 2002:** Greene, S.L. Characteristics of applications that support creativity. In journal Communications of the ACM volume 45, 2002.
- **Guilford 1950:** Guilford J.P. *Creativity Research: Past, Present and Future*. American psychologist, 1950.
- **Hasting 2009:** Hastings, E.J. and Guha, R.K. and Stanley, K.O. *Evolving content in the galactic arms race video game*. Proceedings of the IEEE Symposium on Computational Intelligence and Games, 2009.
- **Hindus 2001:** Hindus, D. and Mainwaring, S.D. and Leduc, N. and Hagström, A.E. and Bayley, O. *Casablanca: designing social communication devices for the home*. Proceedings of the SIGCHI conference on Human factors in computing systems, 2001, pp. 325-332.
- **Holland 1975:** HOLLAND, J. H. *Adaptation in natural and artificial systems*, 1975.
- **Landay 1995:** Landay, J.A. and Myers, B.A. *Interactive sketching for the early stages of user interface design*. Proceedings of the SIGCHI conference on Human factors in computing systems, 1995.
- **Lee 2010:** Lee, B. and Srivastava, S. and Kumar, R. and Brafman, R. and Klemmer, S.R. *Designing with interactive example galleries*. Proceedings of the 28th international conference on Human factors in computing systems, 2010, pp. 2257--2266
- **Lin 2000:** Lin, J. and Newman, M.W. and Hong, J.I. and Landay, J.A. *DENIM: finding a tighter fit between tools and practice for Web site design*. Proceedings of the SIGCHI conference on Human factors in computing systems, 2000.
- **Masson 2010:** Masson, D. and Demeure, A. and Calvary, G. Magellan, an Evolutionary System to Foster User Interface Design Creativity. In proceedings of EICS'10, Berlin, 2010.
- **Monmarché 1999:** Monmarché, N. and Nocent, G. and Slimane, M. and Venturini, G. and Santini, P. *Imagine: a tool for generating html style sheets with an interactive*

genetic algorithm based on genes frequencies. In Proc. of IEEE Intl. Conf. on Systems, Man and Cybernetics, 1999.

- **Myers 2000**: Myers, B. and Hudson, S.E. and Pausch, R. *Past, present, and future of user interface software tools*. Journal Transactions on Computer-Human Interaction, 2000.
- **Oliver 2002**: Oliver, A. and Monmarché, N. and Venturini, G. *Interactive design of web sites with a genetic algorithm*. In Proceedings of the IADIS International Conference WWW/Internet, 2002.
- **Paterno 1997**: Paterno, F. and Mancini, C. and Meniconi, S. *ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models*. In Proceedings Interact'97, July'97, Sydney, Chapman&Hall, 1997, pp. 362-369.
- **Plessis 2008**: du Plessis, M. C. and Barnard, L. 2008. *Incorporating layout managers into an evolutionary programming algorithm to design graphical user interfaces*. In Proceedings of the 2008 Annual Research Conference of the South African institute of Computer Scientists and information Technologists on IT Research in Developing Countries: Riding the Wave of. SAICSIT '08, volume 338. pp 41-47.
- **Quiroz 2007**: Quiroz, J.C., Dascalu, S.M., and Louis, S.J. *Human guided evolution of xul user interfaces*. In CHI '07 extended abstracts on Human factors in computing systems (New York, NY, USA, 2007), ACM, pp. 2621-2626.
- **Quiroz 2009**: Quiroz, J. C., Louis, S. J., Banerjee, A., and Dascalu, S. M. 2009. *Towards creative design using collaborative interactive genetic algorithms*. In Proceedings of the Eleventh Conference on Congress on Evolutionary Computation. IEEE Press, Piscataway, NJ, 1849-1856.
- **Secretan 2008**: Secretan, J., Beato, N., Ambrosio, D.B., Rodriguez, A., Cambell, A., and Stanley, K.O. *Picbreeder: evolving pictures collaboratively online*. In CHI '08: Proceeding of the twentysixth annual SIGCHI conference on Human factors in computing, 2008, pp. 1759-1768.
- **Seffah 2005**: Seffah, A. and Gulliksen, J. and Desmarais, M. (2005). *Human-centered software engineering*. 2005, Springer.
- **Simonton 1999**: Simonton, D. K. *Creativity as blind variation and selective retention: Is the creative process Darwinian?* Psychological Inquiry, 309-328, 1999.
- **Sims 1991**: Sims, K. *Artificial evolution for computer graphics*. In SIGGRAPH'91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1991), ACM, pp. 319-328.
- **shneiderman 2007**: Shneiderman, B. *Creativity support tools: accelerating discovery and innovation*. Journal Communications of the ACM, volume 50, 2007.
- **Stanley 2002**: Stanley, K.O. and Miikkulainen, R. *Evolving neural networks through augmenting topologies*. Journal Evolutionary Computation volume 10, 2002.
- **Szekely 1993**: Szekely, P. and Luo, P. and Neches, R. *Beyond interface builders: Model-based interface tools*. In proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems, 1993.
- **Szekely 1994**: Szekely, P. *User interface prototyping: Tools and techniques*. Software Engineering and Human-Computer Interaction, 1994.
- **Szekely 1996**: Szekely, P. *Retrospective and Challenges for Model-Based Interface Development*. Proc. of 3 rd Int. Workshop on Computer-Aided Design of User Interfaces CADUI'96, 1996.

- **Suwa 1999:** Masaki Suwa, J. and Gero, J. and Purcell, T. *Unexpected Discoveries And S-Invention Of Design Requirements: A Key To Creative Designs*. University of Sydney, 1999.
- **Takagi 2001:** Takagi, H. Interactive evolutionary computation as humanized computational intelligence technology. In *Fuzzy Days (2001)*, p. 1.
- **Tohidi 2006a:** Tohidi, M. and Buxton, W. and Baecker, R. and Sellen, A. *Getting the right design and the design right*. Proceedings of the SIGCHI conference on Human Factors in computing systems, 2006.
- **Todhidi 2006b:** Tohidi, M. and Buxton, W. and Baecker, R. and Sellen, A. *User sketches: a quick, inexpensive, and effective way to elicit more reflective user feedback*. Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles, 2006.
- **Virzi 1996:** Virzi, R.A. and Sokolov, J.L. and Karis, D. *Usability problem identification using both low-and high-fidelity prototypes*. Proceedings of the SIGCHI conference on Human factors in computing systems: common ground, 1996.
- **Warr 2005:** Warr, A. and O'Neill, E. Understanding design as a social creative process. In *Proceedings of the 5th conference on Creativity & cognition*, 2005.