Proactive User Interfaces for Cultural Resource Management Systems

César A. González Pérez

Grupo de Trabajo en Arqueología del Paisaje.Universidade de Santiago de Compostela

Apdo. de Correos 994. 15700 Santiago de Compostela. Spain

E-mail: phcgon@usc.es

Introduction

This paper shows the results of work on database user, interface design and construction, that the *Grupo de Trabajo en Arqueología del Paisaje* has developed, during the last two years. We believe that the generic rules and principles, that emerge from this work, can be applied to the design and construction processes of any user interface.

Basically, user interfaces are necessary, due to the need for communication between *machines* and the people, working with these machines, the *users*. Thus, every system running on a machine (be it a computer, or not) needs an appropriate user interface, as a meeting point for both the machine and the user.

Most current systems find this meeting point half-way between the two, making people move in order to come closer to the machines. This process of coming closer usually requires too great effort, thus, leading to hardly usable systems. The basic reasons for low usability are the limited possibilities for information transfer, which a user is able to perform, and also the imposed necessity for advanced knowledge of the reality, modeled by the system, as well as the criteria, used for the creation of the model.

The Conventional Case

A good example is that of a user interface employed to query a standard, relational database. A medium-complexity database, perhaps including more than fifty or sixty tables, models a segment of reality, following criteria that the database developers considered, during its design. Facing a window, that shows a list of available tables, a user needs to know that reality, as well as the modeling criteria, in order to pose a query, that makes sense in the context of such a database.

Specifically, we have observed in our group, that building queries in relational databases, by using user interfaces such as the one described, constitutes a high-effort, low-productivity task.

Modularity and Proactivity

On the other hand, a proactive, user interface would show the system to the user from a modular viewpoint, decomposing it harmoniously with the observed reality that it models, and therefore, allowing the user modular interaction.

Also, with such modularity as a base, a proactive, user interface would clearly mark, at each instant, the neighbourhood of the module, with which the user is interacting, offering semanticsof both spatial and temporal *immediateness*. This performance can be described as immediate interaction.

Further, a proactive, user interface would conceal those parts of the system, not readiliy available to the user, from the module with which he or she is interacting, at each instant, only showing information related to the context of the interaction, and allowing, therefore, contextual interaction. In fact, contextual interaction constitutes the fundamental principle of proactive, user interfaces.

A proactive, user interface, built around the shown principles, certainly implies a change of attitude for system users. In conventional systems, interaction is attained by the user giving the system a big amount of information, to which the user interface responds minimally. Conversely, systems with proactive user interfaces require less initial specifications, by the user, and the contextual information, which they return, is much richer. In fact, *the proactivity of a proactive, user interface stems from its ability to act, before the user, to delimit the working context.*

An example of a proactive, user interface could be a system, in which, in order to pose a query to a database, the user must first choose a context, for which he or she needs information, from global list. Once this context has been specified, the system can deduce in which relationships, such contexts take part, and shows a list containing them. After allowing the user to select one, the system shows a new list with those contexts, that can be involved in the chosen relationship. We must emphasize that the second context list does not show all the contexts in the system, but only those, that can be accesed from the initial one by following the chosen relationship.

Properties

Such a system must have two fundamental properties: first, it must offer a good representation of the observed reality that it models. A *good* representation is one, which is faithful to reality, and which does not introduce any concepts, foreign to it. Some factors that may support a good representation are a highly expressive modeling environment and a correct and verifiable modeling process.

Second, the *structural semantics* describing the observed reality, modeled by the system (i.e., information about the

structure of the system), must be available to it. Thus, the system must be highly self-descriptive.

Object-oriented technologies provide an advantageous way to obtain the good representation that we need, given that their expressive power is much higher than that of classical development paradigms, such as the procedural or relational ones. Simultaneously, object-oriented technologies make the modeling process easier, both at the technical (seamless transition between development stages, reusability) and organizational (evolutionary life-cycles, reusability) levels.

On the other hand, the availability of the system's structural semantics can be achieved by using meta-information, that is, *information comprised of referents, the substrates of which are objects, internal to the system.* Generally speaking, a referent is an object, that represents another object, and a substrate is the represented object. Conventional systems are sets of referents, the substrates of which are found outside the system, that is, in observed reality. A system with meta-information, on the contrary, must contain referents to internal substrates.

Thus, such meta-information is found within the system, and is, therefore, available to it by conventional means.

Meta-information

A meta-information-based, user interface is, therefore, capable of using the description, that the system offers about itself, to give the user an appropriate view of the system (considering an appropriate view to be one which is faithful to the observed reality). This allows high knowledge transfer from reality to the system, to be performed by the user, resulting in a more usable system.

In addition, the user interface is not hard-wired (following more or less specific criteria), but dynamically generated by the system; such a user interface is much more modular, providing better extensibility, reusability and robustness.
However, using meta-information-based, user interfaces also poses some problems, such as the increment in the number of abstraction levels, managed by the system. We have found that this factor, called *system depth*, is what most notably affects its final complexity.

At the same time, some simplifications, often assumed during the design and construction of conventional systems (such as those concerning the departure of the system from observed reality), become impossible, or very difficult.

Finally, these factors produce an increased system complexity, which must be taken into account from the start. Nevertheless, we believe that the benefits of using object-oriented technologies, strongly outweigh the disadvantages of using meta-information.

Systems Modeling

In a classical life-cycle, the development of a conventional system goes through four main stages. Generally, the ease of transition, from one stage to the next, decreases as the process continues, resulting in easier early stages, and harder late ones. The user interface often requires a great amount of work and resources, and is usually implemented as a derivation of the implementation of the rest of the system, sometimes reflecting details, that take it far from the observed reality it models. Because of this, the fidelity of the user interface, regarding system requirements, is low, as is the usability of the resulting system.

During the development of an object-oriented system, conversely, transitions between early stages are more difficult, but the later ones are easier. At the same time, a meta-information-based, user interface can be easily built, using the system analysis (i.e., the system structural semantics) as a starting point. This results in a higher fidelity for the user interface, with regard to system requirements, and, subsequently, a higher usability.

Some Examples

The IPEM (Meta-information-based Experimental Proactive Interface) prototype shows a user interface, for object-oriented database querying. This system uses an object-oriented, database simulator, running on a relational engine (because we have not yet found on the market, any system of good quality, with good price and usability factors).

The IPEM prototype provides the user with contextual interaction, showing only those parts of the system which can be directly accesed from the current working context at each moment in time. It provokes a change of attitude, in the user, who feels relieved by the help that the system gives him or her.

The R&D project, called CRISys (project code CICYT TEL96-1386), is aimed at building a computerised framework for Cultural Resource Management.

CRISys is totally based on object-oriented technologies, and includes a database system with meta-information. The user interface has been conceived, as one more sub-system in the whole system, and is being implemented as a generic user, interface engine, parameterizable through the structural semantics of the host system, and capable of showing, to the user, any self-descriptive object in the system.

Consequences

Before moving further into exploring the field of user interfaces, we believe that it is necessary to adopt the appropriate methods and tools, required to obtain a high-quality modeling process, that is, to fully commit to object-oriented technologies.

This would support a good correspondence between observed reality and the user interface, of computerised systems, which in turn, would allow us to base such user interfaces in the meta-information, contained in the same system.

The ultimate consequence of using meta-information is the capability of contextual interaction (or proactivity) by the user interfaces, a fundamental characteristic in vertical, affordable, and commercially successful systems.