A multimedia system for archaeological scenes

Chrystalla Alexandrou, George A, Papadopoulos
Dept of Computer Science, University of Cyprus, Nicosia, Cyprus
E-Mail: cschryst@turing.cc.ucy.ac.cy, george@turing.cc.ucy.ac.cy

Reem Bahgat
Department of Computer Science and Information Sciences
Cairo University, Cairo, Egypt
E-Mail: rbahgat@ritsec.com.eg

1 Introduction

Archeologists and historians have now the ability to use computers and software in order to document as well as use the documentation in their area of work. The method that is being used by most of the systems is the documentation via a free text that at best is enriched by hypertext links and codes. The documentation of pictures and photographs is done under a separate system of videodisks or photothekes. Some other systems are using a relational database approach. Both the approaches have major disadvantages: the first one is rather limited in manipulating the information stored and also this information is divided according to the media used for storing; the second one is unable to represent the wide range of types, the continuous schema evolution and the complex relationships that the system requires.
We propose instead a multimedia Object Oriented (OO) approach since archaeological systems require wide range of types because of the significance in the details of the data representation, that the OO can serve the best way via inheritance, composition and polymorphism. Also a multimedia OO system can store under the same system different media types, something that assists the application in being more accurate and specific (the best description of an image is the image itself!). As a state of the art example we have started to build an OO system for the documentation and data retrieval for archeological scenes.

2 Proposed system

We can state that a scene is a composite object that has important passport data like the date of the creation, the materials that are used, its location etc. On the other hand, what is more interesting, is to document what this scene represents: the different entities in the scene as well as the relationships between them. The system must satisfy queries at three different levels: a. Queries on the scene passport data (like find all the scenes that are made of stones in Egypt) b. Queries on the scene elements passport data (like find all the scenes that contain a king). c. Queries on the elements relationships. (like find all the scenes that have a king holding a knife).
The information gathering starts with the digitization of a photograph of the scene and the filling up by the documentor of a form that requires the passport data of the scene. The identification of the different important elements of the scene is done via two L axes that the documentor can move in the digitized image in order to define the minimum rectangle that contains the element. All the different categories of elements are provided by the system in different class hierarchies and the documentor has to choose the right class for the object that is going to describe. The relationships between the different elements are documented again via predefined types of relationships like a "hold" relationship, a "wear" relationship, etc.
It is obvious that the system is very strict at the definition level. The documentor always classifies the data in predefined classes and completes the requirements for the specific instance, mostly by using of dictionaries. The documentor has the ability to enrich the dictionaries, if he has the privilege to do so. We want to extend the schema evolution, that the OO database can provide to the system maintenance, in order to give to the documentor (user) the ability to "transfer" an existing description through the hierarchy. Which means, that the hierarchical structure must also be accessible to the documentor since in the manual system classification is one of the main duties of the archaeologist. From a graphical view of the class hierarchy the documentor has the ability to navigate into the hierarchy and choose a new class and the system will automatically transfer the required data and prompt the user to complete the requirements of the new type. Our hierarchies start from the same root Tobject class and this is giving to the user the chance to change any class with the associated cost.