International Journal of Recent Engineering Science (IJRES),
ISSN: 2349-7157, *volume1 issue3 May to June 2014*

Using MVC Pattern for Reengineering Medical Application

R.Sowbarnica , Dr.S.Niraja

Research Scholoar , Assisstant Professor
Department of computer Science,
Hindusthan College of Arts and Science, Coimbatore

Abstract

Medical Imaging is one of the innovative research areas dealing with storage, retrieval and processing of medical images. With the tremendous technological growth in storage devices and algorithms, the research focus is now on the developments of flexible user interfaces for the physicians who frequently interact with the system for diagnostic purposes. During the diagnosis process, they face many problems in remembering the processing steps and parameters used by them. The steps involved in the processing might constitute the *Diagnostic Protocol* if they proven to be consistent across many cases. Such protocols have to be recorded conveniently for efficient reuse. This paper aims at providing an architecture that will automatically register the diagnosis steps as hyperlinks that may be used as an alternate way of triggering the applications without having the need to remember the steps. The pattern Model/View/Controller has been effectively used for the development of the architecture.

KEY WORDS: Protocols, Hyper-Control, Design Patterns, Model/View/Controller, Object oriented metrics.

1. INTRODUCTION

A *Hyper Control Document* is a multimedia document that might be used to activate an existing stand-alone application with the necessary request associated with some parameters received from the users. The stand-alone application being controlled by a hyper control is known as a *Hyper-Controllable Application System*.

Physicians have to carry out a lot of image processing operations as a part of their diagnosis. They use the standard user interface comprising of menus, tool bars and keyboard shortcuts. For a complicated system, it becomes difficult to remember these commands, their sequences and associated parameters. So they need an alternate mechanism of reusing the previously executed commands and parameters, if any used. If the same procedure adopted by them consistently produces reasonable results, then it has to be saved as a *Diagnostic Protocol* [7] for the particular case. When these protocols are registered in a convenient form, they promote reusability, accuracy and speed of operations. Hyper-control is one such mechanism of providing an external control for an existing application without introducing major modifications on the existing code. When many features of hypermedia applications are included in an existing system to increase their utility and usability, a new concept known as "*Hypermedia Functionality*" [6] arises. A hyper media system has an added advantage of integrating the formal (used by the machine) and informal (used by the physician) knowledge representations [4].

Many application designers prefer to use patterns for their development. Patterns can be employed in architectural, design and implementation levels. The architecture suggested in this paper uses many patterns as they provide efficient coordination between objects by encouraging the principles of object-oriented design namely '*Program to an interface, not an implementation*' and '*Favor object composition over class inheritance*' [1].

This paper is organized as follows: A few related works has been presented under section 2. Section 3 explores the proposed architecture Hyper Control for Medical Experts (HCME). Section 4 contains the performance analysis of HCME. Section 5 concludes with some future directions.

2. RELATED WORK

The **CAIBCO** System [9] is a web-based object-oriented multimedia medical system which combines the two technologies for a better interaction by the students and medical staff spread over wide geographical areas. However, it doesn't include patterns in the development process of the system.

G. Rossi et al [5] describe some patterns which may be useful for the reuse of design in an hypermedia application. An approach has also been described for the development of a pattern language for hypermedia. The pattern systems illustrated here are designed to work only with the hypermedia systems. The patterns may not be suitable for providing an external hyper-control for an existing non-web based application.

DISCOVER [7] is a distributed interactive visualization system aims at providing an external control to an existing application. Five different patterns are used for the implementation of the application.

The patterns used in DISCOVER namely Command Processor, Memento, Document-View, Visitor and Singleton are all designed to be inter-dependent. In other words, it improves tight coupling between some of the classes defined by the patterns, which is not desirable in pattern-oriented programming. Similarly, the command processor is duplicated for independent handling of dialog codes and computation codes which doubles the generation of command objects, which may result in inconsistency.

There are also other side effects in the patterns used in DISCOVER. Command Processor introduces excessive number of command classes that may result in efficiency loss and there may be complexities in acquiring command parameters [2]. The pattern Visitor is less helpful during growth stage since every new class getting added in the Visitor's list has to update entries in the abstractVisitor class and concreteVisitor class[3]. In general, Visitor is not flexible when changes are possible in an application. The Memento may not be suitable for image processing applications as the amount of data that a Memento must save might be quite large and taking up a fair amount of storage space [3].

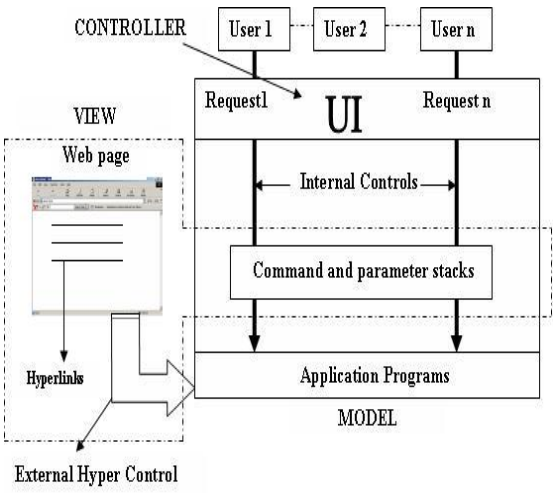
3. ARCHITECTURE OF HCME

The objective of HCME is to provide an external control for an existing application by automatically generating the HTML codes from the user invocations. Web pages are created for the operations carried out by the physicians using internal control. The architecture is constructed using a different combination of patterns in order to avoid the problems discussed under DISCOVER. HCME has been tested on a set of stand-alone image processing programs. The programs can be triggered successfully using hyper-control with three different types of invocations.

- Programs that do not require any parameters during invocations (DIRECT).
- Programs that automatically work on previously used and saved parameters (REUSE).
- Programs that work after receiving the necessary parameters from the user (GET).

HCME consists of a) user interface (UI) b) stand-alone application programs to carry out the image processing activities c) two stacks for storage, one for the commands issued by the physicians and another for the parameters passed by them and d) two controllers to coordinate the dialogues and triggering of applications.

Physicians use the UI part to send their request along with the parameters (if any). These internal controls and the parameters used in invoking them are trapped and made available in the command stack and the parameter stack respectively. This forms the basis for the external control. The external control makes use of this information and generates hyper controls automatically, that are subsequently stored as web pages. These controls provide an alternate way of invoking the application programs in addition to the available internal controls. When the user selects the hyper controls, the corresponding programs are executed. This is illustrated in Fig(1).



Fig(1): The External Control Mechanism

The three invocations discussed earlier are dealt differently. If an invocation calls a program of type DIRECT, it requires no parameters and executed directly. If it is of type REUSE, the programs are executed with previously used parameters which are 'poped' from the parameter stack and if the program belongs to the type GET, then the required parameters are obtained interactively.

Five different patterns are employed in the development of the architecture. The pattern *Command Processor* [2] has been used to provide controlling for the computation and dialog codes. *Memento* [1] pattern is used for saving the states of command objects for future use. *Observer* [1] pattern takes the role of checking the state change every time and announces the *Memento* to go for appropriate actions and *Strategy* [1] decouples the requests from the invocations. The *Model/View/Controller* [2] has been used as the architectural pattern where the *Model* components deal with the business logic, *View* generates the required web pages and *Controller* keeps track of invocations, stores them in a stack.

The implementation of the architecture was carried out in top-down fashion. During the first step, the responsibilities of the application were classified under the *Model/View/Controller* components. The *View* was designed to hold the generated web pages, the *Model* to hold the stand-alone applications and the *Controller* was made to store all the interactions made by the physicians. During the second step, the contexts for the applications of more patterns were identified.

4. PERFORMANCE ANALYSIS

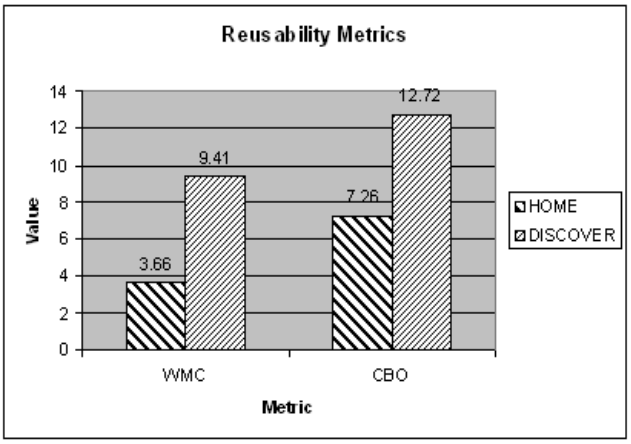
Java 2 Enterprise Edition (J2EE) ™ was used for the implementation of HCME. Retinal Fundus images in JPEG format of size 12K to 15K are used for testing.

4.1 Reusability Metrics

The two reusability metrics involved are *Weighted Methods per Class(WMC), and Coupling Between Objects (CBO)*[8].

WMC gives the count of the methods implemented within the class. Fig(2) indicates the WMC values obtained for HCME and DISCOVER are 3.66 and 9.41 respectively. The lower value for HCME indicates that a sign of improvement in reusability, as classes with smaller count of methods will be less application specific [8].

The metric CBO denotes the number of other objects invoked whenever a particular object is invoked. From Fig(2), the CBO value for HCME is 7.26 against that of 12.72 for DISCOVER. This lower value for HCME shows the reduction in dependency between objects [8], which is again a sign of promotion of reusability.



Fig(2): Chart showing comparison of values obtained for reusability metrics

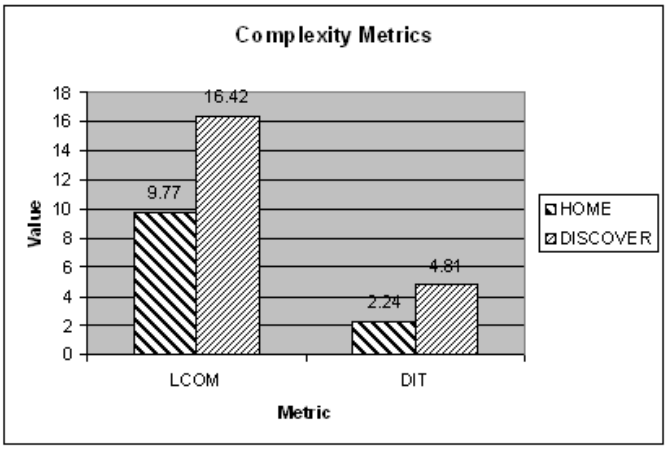
4.2 Complexity Test

Lack of Cohesive Methods (LCOM) and Depth of Inheritance Tree (DIT) [8] are the metrics used for evaluating the complexities of the architectures.

The LCOM value corresponds to the average number of dissimilar methods of a class. Existence of cohesive methods indicates proper sharing of responsibility among classes. A high LCOM value indicates increase in complexity and difficulty in maintainability. When there are more cohesive methods, the LCOM value, which is the complement of the fact, has to be low. Fig(3) gives the values of LCOM for HCME and DISCOVER as 9.77 and 16.42 respectively. So, the low LCOM value for HCME indicates a reduction in complexity.

DIT corresponds to the average depth every class has from its first level parent class. The low DIT value indicates a decrease in complexity since this reduction corresponds to the average depth of a class from the root

node. From Fig(3), the values of DIT for HCME and DISCOVER are 2.24 and 4.81 respectively. Even though the DIT value for HCME is favorable, there is no much significance in HCME when compared with that of DISCOVER.



Fig(3): Chart showing comparison of values obtained for complexity metrics

4.3 Patterns Used & Their Consequences

Memento: Memento is used with a purpose of saving the state for future use of undo/redo functions. However, as discussed earlier, the amount of storage for applications like image processing is very high and this may introduce a large amount of memory utilization and which may result in inconsistency. So, the Memento implementation is designed to store only incremental changes during successive invocations. This is possible only when Memento is collaborated with Observer pattern. The modified class of memento takes up the code of the following form.

```
public void setMemento(Memento temp)
{
  .........
  .........
   // If any Observable changes occur then
only a new
  // Memento object is created
  if(mobserv.sendNotify(m[(t-
1)].getImage(),temp.getImage()))
  {
    // Code for creating Memento
  }
  ............
  ............
}
```

Observer: Observer collaborates with Memento in saving the objects only when a state change is 'observed'.

Strategy: This pattern replaces the Visitor of DISCOVER. This pattern is ideal for software systems with frequent additions. Moreover, Strategy works on object decomposition, which is a more powerful way of extending a class's functionality when compared with class inheritance.

5. SUMMARY AND CONCLUSION

Hyper-control is a powerful and flexible mechanism to standardize the processing procedures. It can make a general-purpose application easy to use in different domains. The basic requirement for an application to support the hyper-control mechanism is to provide an additional input channel to receive external control from other applications. This work extends the Command Processor pattern by separating the computation codes and dialog codes into two kinds of command objects, i.e. the Dialog and Computation command objects, to support the hyper-control mechanism.

Besides, when combined with Document-View pattern, a command object may have different supplier types (image document types). In such a case, the reusability of a command object becomes an important issue. By adopting Visitor pattern for execution and Memento pattern for the undo function, a command object can be applied to different supplier types without any change.

In order to receive requests from hyper-control documents or other applications, two proxy controllers are also provided. This allows hyper-control documents to become part of the application's user interface and to be integrated with an HTML-based help system easily.

Based on above pattern technology, an interactive medical image application system has been developed to support the hyper-control mechanism successfully. During the development process, it was discovered that patterns do provide a very good solution in the design of system architecture to solve the practical problems. Although several patterns work together, they still help to have a clear road map of the complicated control flow. Our future work involves discovering a pattern framework for automating medical applications.

REFERENCES

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, "Design Patterns – Elements of Reusable Object-Oriented Software", Addison-Wesley.

[2] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, (2001) "Pattern-Oriented Software Architecture", Volume 1, John Wiley & Sons (Asia) Pte Ltd.

[3] James W. Cooper "Java Design Patterns" Pearson Education, Inc.,

[4] Daniel Schwabe, Gustavo Rossi, (1995) "The Object-Oriented Hypermedia Design Model (OOHDM)", Communications of the ACM.

[5] Gustava Rossi, Daniel Schwabe, Alejandra Garrido,(1997) "Design Reuse in Hypermedia Applications Development", ACM.

[6] John M. Vlissidess, James O. Coplien, Norman L. Kerth, (1998) "Pattern Languages of Program Design", Addison-Wesley, 4th printing.

[7] Ku-Yaw Chang, Lih-Shyang Chen, "Using Design Patterns to Develop a Hyper-controllable Medical Image Application", PLoP-98.

[8] Linda H. Rosenberg, (1998) "Applying and Interpreting Object Oriented Metrics", Proceedings of the Software Technology Conference, Utah.

[9] Miguel Aluen, Hector Arrechendera, Alfredo Matteo, Christiane Metzner, (1999) "Developing a Web-based Object-Oriented Multimedia Medical System", Proceedings of the 32nd Hawaii International Conference on System Sciences.