Software Architectural Design for Image Retrieval System involving Data Streams and User Interactivity

Milu Mary Philip, B.Vijayakumar

***Abstract*—In the process of software development, the role played by the software architect is substantial. It is the software architect who splits the entire application into different components, understands the functions performed by each component and should possess sound knowledge on the interactions among these components. The software architect communicates this information to the developers of the software system. The expectations and requirements of a user change often during the initial phases of system definition and development. With each change in the requirements of a user, the architectural design of the complete software system undergoes frequent changes. In this paper, we propose a software architecture in which the structure and behavior of the system can be changed at any stage of the software development life cycle. The retrieval of images is considered a typical scenario in the current work. The proposed architecture considers a combination of modes pertaining to data streams and user interaction.**

***Index Terms*— Software Engineering, Software Architecture, Architectural Patterns - Model View Controller, Pipes and filters, Reflection.**

I. INTRODUCTION

WITH the increase in the size and complexity of a software system, the process of designing and specifying complete software architecture turns out to be an increasingly complex task. Software architecture includes the description of elements from which systems are built, interactions among them, patterns that guide their composition and the constraints on these patterns [1] [2]. The software architect has the responsibility to propose a suitable architecture for an application based on the requirements, constraints and quality attributes [3][4]. The major responsibilities of the software architect include the following- the splitting of a complex application into constituent parts, understand the functions of each component in the application, to understand the interactions and dependencies among the components and communicate these concepts to the developers.

Manuscript received July 11, 2014; revised August 06, 2014.

Milu Mary Philip is with the Department of Computer Science, Birla Institute of Technology and Science, Pilani, Dubai Campus. E-mail : p2012003@dubai.bits-pilani.ac.in.

Dr. B.Vijayakumar is with the Department of Computer Science, Birla Institute of Technology and Science, Pilani, Dubai Campus. E-mail : vijay@dubai.bits-pilani.ac.in

The present work proposes a mechanism for changing the structure and the behavior of a software system at any stage of the software development cycle. The development and maintenance costs of an application system can be reduced considerably by supporting the variation points. A combinational approach is followed for dealing with both data streams and user interactivity. This paper is organized into six sections namely Introduction, Objectives, Motivation, Related Work, Proposed Architectural Design for Image Information Retrieval and Conclusion.

II. OBJECTIVES

The proposed work aims at designing a suitable architecture for interactive and data-stream-oriented systems. These systems are characterized by changing requirements during the requirements analysis and design phases. The specific objectives are stated as follows:

- Provide a mechanism to specify, manage and store changes in the user requirements.
- Propose a flexible architectural design for a software system involving both user interactivity and data-stream mode of operations.

III. MOTIVATION

Software architect faces a number of challenges in designing a suitable architecture for applications with changing requirements [5]. Changes in user requirements can occur during the definition, development and maintenance phases of software systems. The module structure, its hierarchy and the various module connections can vary for each application and also within a single application. Frequent changes in requirements can result in increased development and maintenance costs of an application. The reflection architectural pattern provides a mechanism to accommodate changes in requirements.

Most of the present-day software systems cannot be designed with a single architectural pattern because different user requirements cannot be handled by a single architectural pattern. Different system requirements can be addressed by a combination of different architectural patterns. A combination of data streams and interactive operations is required for present-day applications. This paper deals with an application that uses both user interactivity and data stream modes of operation. The model view controller (MVC) pattern has been chosen for user interactive system and the pipes and filters (P&F) pattern for data-stream mode of operation.

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

IV. RELATED WORK

This section gives an overview of existing literature on architectural patterns, different architectural styles like pipes and filters, model-view-controller and reflection patterns. As the work deals with an application that involves both data stream as well as interactive modes of operation, this section deals with a brief review of few user interactive applications and those that work in data stream mode.

Software architecture is important and essential in the development of a large and complicated system. It is considered an effective solution to functional and quality requirements [3][4]. There is a close relationship between designing the architecture of a system, architecture styles used and the quality properties [6]. Software architectures with well-defined properties can be constructed with the use of patterns [7]. The success of the pattern selected is dependent on how it meets the objectives of software engineering. The development of a software system with well- defined properties is possible with the help of right pattern selection.

During the process of designing software, a software architect depends a lot on architectural patterns [5]. For an architect to choose the right architectural pattern, the interaction between quality attributes, tactics and patterns should be analyzed. Selecting the correct pattern is of great importance because these styles impact the characteristics of the software [8].

The pipes and filters architectural pattern (P&F) is used in applications where data streams can be processed incrementally[2],[7]. It is not ideal to implement such a system as a single component as the requirements are never constant and also because the system has to be built by different developers. The entire task of the system is hence divided into different processing steps, each of which communicates through the data flow between the systems. Each of these steps would act as filters and the connectivity is brought out through the pipes [9][10].

The multimedia framework consists of many media processing components [11] in which data flows in streams. Hence the software architecture of such a system uses the pipes and filters architectural pattern. Streaming graphs are built by the frameworks, where the nodes act as the filters that perform specific functionality and the edges of the graph would be the pipes that connect the media components.

U.Banodha and K.Saxena [10] in their work, uses the P&F pattern in the field of medical process reengineering. The security feature was also incorporated to different processing stages in the work by Eduardo B Fernandez [12].The chances of reusability and flexibility is also increased by allowing different combinations of the processing modules in the P&F pattern [13].

Human-computer interaction is made possible with the help of graphical user interfaces. The Model View Controller (MVC) architectural pattern is the best known design for such systems [2]. The entire application is divided into model, view and controller components [14-16]. The user interface would be the view component through which the information would be displayed to the user. Inputs from the user would in turn be passed to the application system through the view component. User entries in the interface would be handled by the controller component [17]. The component that holds the complete data of the system would be the model component and it also provides various functionalities of the system.

The MVC pattern has been deployed in a document processing software [18] where the data access, data presentation and the data processing tasks are separated into different components. The complexity of managing the contents of the documents is also minimized with this approach [15]. In the case of a web application scenario [19], the model component resides on a server and is accessed using the hypertext transfer protocol. The entire communication between the client and server is carried out by the controller component.

The pattern system can also be applied in ATM Simulation Systems [14] for banking applications, online examination system [20], in the development of information systems in film production processes [21]. It is observed that this pattern helps in faster development of the system and also enhances system usability. Tabacu Iulia in their work [22] have used the MVC pattern to create a component that will simulate an animated view of the images of the different TV channels.

Software testing divisions also use the MVC pattern for the development of a test suite design [23]. A test suite library which consists of all the tests would be the model, the user interface through which each user can modify the test cases and the controller which would obtain the requirements of the test member.

Software architecture of a system should be designed in such a way that it can handle the changing requirements of the user. This is made possible using reflection architectural pattern where the entire application is sub-divided into meta level, base level and a meta object protocol. Information about the software is provided by the meta objects present in meta level. The services available at the meta level consists of – method definition, field assignment and its access and modification of function call mechanisms [4],[6]. Application logic is being dealt with the base level and hence is independent of changes in user requirements. The third level called the meta object protocol supports the implementation of functions operating on the meta objects.

A combination of different architectural styles is required for the development of large scale software systems. The distributed military troop deployment and battle simulation system (TDS) [24] is a large scale system where a combination of four different architectural styles like component and message based, client-server, peer-to-peer and push-based have been used. Yet another example is the Attention System with Multiple Camera (ASMC), which involves the combination of push-based, peer-to-peer and pipes and filters architectural styles [24].

The Unified Software and Hardware Architecture (USHA) has been proposed [25] and hence flexibility is achieved so that a large variety of video formats can be handled simultaneously. Software architecture for supporting multi-touch capability has also been developed [26] which unifies the different input methods and reduces the need for customized applications.

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

V. PROPOSED ARCHITECTURAL DESIGN FOR IMAGE INFORMATION RETRIEVAL

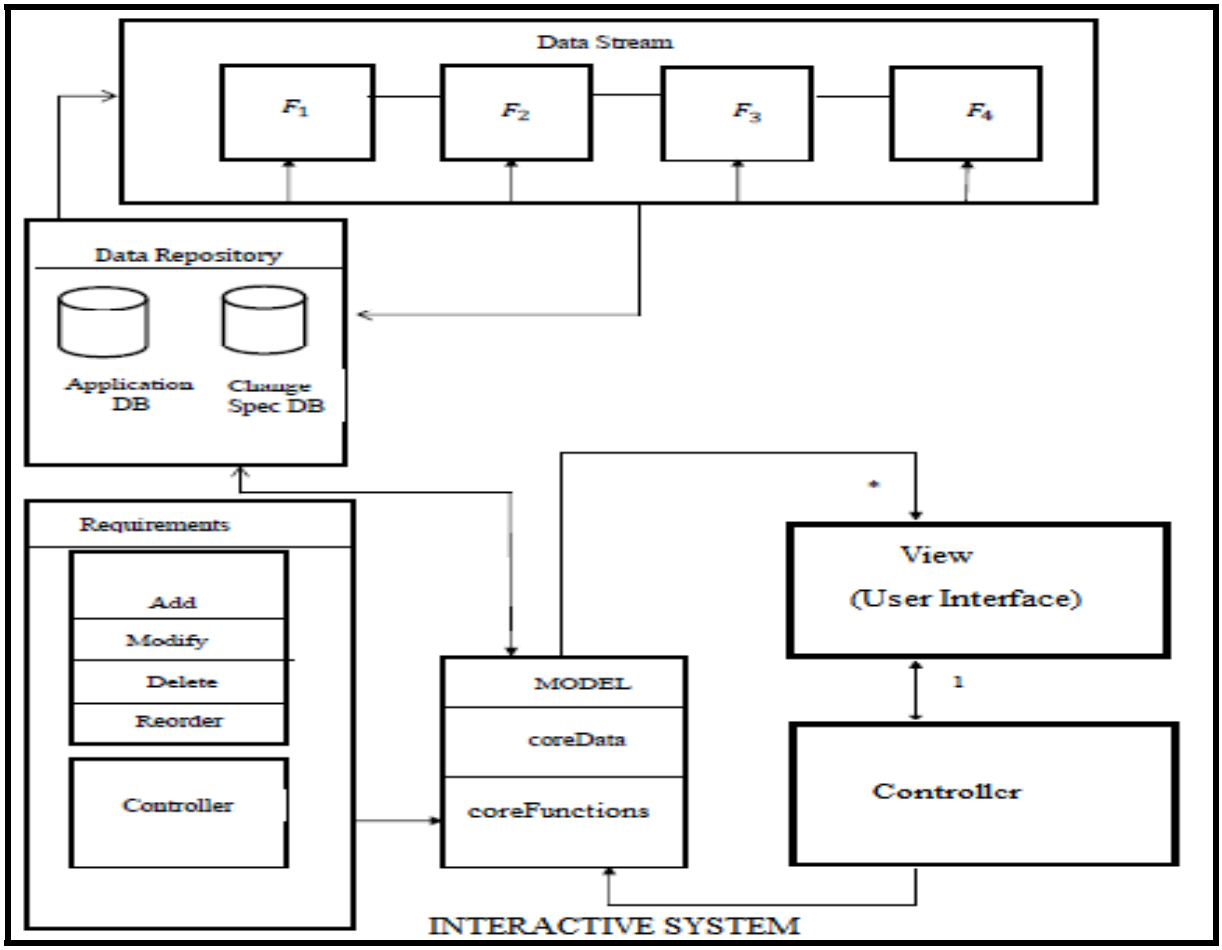

Fig 1 : Architecural Design for Information Retrieval of Images

This section focuses on a practical application for information retrieval of images. This application requires a flexible architecture involving data streams and user interactivity. Flexibility is incorporated by allowing the user to interactively specify input parameters for the images, required modules for image processing and their processing sequence. Fig 1 shows the architectural design for this application.

The application involves retrieving one or more images from an image database for a given keyword search. The keyword search is given in the form of queries over images. A typical query to retrieve an image whose textual details would have the required keyword in it is as shown below:

"Retrieve all images from the image library in which the textual details of the images matches with the given keyword"

All the images matching the keyword will be retrieved from the database and passed on to the model component of the interactive system. For the image retrieval scenario that is described below, the *coreData* of the model component refers to the set of images that are retrieved from the database and the *coreFunctions* refers to the different procedures that can be used for processing of the data such as the selection of 2D/3D images, color/black and white images. The different steps for a typical scenario is as shown below – it begins with the *interactive system*, followed by the interaction between the *interactive system* and the *data repository*, the communication between the *data repository* and the *data stream* and finally the updated images being displayed in the user interface.

1) Interactive System

- The user selects options for image dimension (2D/3D) and image type (black and white/color) from the list of options (color /black and white, 2D/3D images) present in the view component.
- The controller calls the required procedures of the model component that select the images of user specified image dimension and type from the images present in the coreData of the model.
- If data in the model component is changed or updated, it gives the updated information to all the views attached to it.

The transformed images will then be displayed in the user interface through which the user can view them. User requirements such as *adding new image processing modules*, *modifying*, *deleting* and *reordering* existing image processing modules would be taken as user inputs by the

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

user interface. The user interface facilitates the following actions:

- The user is allowed to select a particular option, say, to add a set of features to the application system – noise removal, image cropping, resizing the image and image format conversion.
- The sequence in which the processing of images is to be performed can also be provided by the user through the Reorder option in the user interface.
- The controller then calls the procedures in the model component to update the *change specifications database.*

2) Interaction between the Interactive System and Data Repository

- Modified images from the interactive system will then be updated in the data repository. The data repository consists of the *application database* and the *change specification database.*

The *application database* consists of a single relation with the following layout:

Images that are retrieved initially from the image database.
Image (**keyword, image file name**, image file size, image format, image dimension, image type, date and time of access).
The primary key is based on two fields keyword and image file name.

The change specification database consists of the following items:

- Feature Details
  Feature_Details (**feature id**, feature name)
  The feature id refers to the module or the components in the data streams.
- Feature Dependencies
  Feature_Dependency (**feature id**, input dependency, output dependency)
  The input dependency for the current feature refers to the immediate preceding component in the data stream whereas the output dependency corresponds to the succeeding component in the data stream. The first component in the data stream will not have input dependency and the last component in the data stream will not have output dependency.

3) Interaction between the Data Repository and Data Stream

- Input data from the data repository is processed in streams. Each processing module will act as a filter. The processed data is passed from one stage to the next stage through pipes.

In Fig 1, filters of the data stream act as image processing modules.

- F_1 – Image processing module of noise removal.
  The noise removal module reduces noises (the unwanted by products while capturing an image). Noise transforms the images by adding extra information to it.
- F_2 – Module for cropping images.
  The term cropping refers to the process of cutting out those parts of an image that are not needed. This is performed to improve framing and to focus on a particular subject matter.
- F_3 – Resizing images.
  Resizing an image can be performed by scaling or resampling it.
- F_4 – Image format conversion.
  Digital images can be organized and stored in different image file formats like jpeg, gif, tiff etc.

4) Interactions between the Data Stream , Data Repository and Interactive System

Processed images will be updated in the data repository and can be viewed through the user interface of the interactive system.

VI. CONCLUSION AND FUTURE WORK

The current work deals with a software architecture that identifies and supports variation points relating to two categories: modules and user interface. This would considerably reduce the development and maintenance cost of a system. Moreover, it would also help a software architect to analyze variation points in the design of new application systems. The proposed architecture provides a mechanism to change the structure and behavior of a software system at any stage of the software development lifecycle.

The present work can be enhanced by incorporating architectural evaluation on the basis of modifiability and correctness quality attributes.

REFERENCES

[1] L. Bass, P.Clements and R.Kazman, *"Software Architecture in Practise"*, 2nd ed. Addison Wesley, Boston, 2003.
[2] M.Shaw and D.Garlan, *"Software Architecture-Perspectives on an Emerging Discipline"*, Prentice Hall Inc, New Jersey, 1996.
[3] E.Majidi, M.Alemi and H.Rashidi, "Software Architecture: A Survey and Classification", in *Second International Conference on Communication Software and Networks ICCSN'10*, pp. 454-460, 2010.
[4] H.J.La, H.J.Lee, and S.D.Kim, "An Efficiency centric Design Methodology for Mobile Application Architectures", in *IEEE 7th International Conference on Wireless and Mobile Computing Networking and Communications (WiMob)*, pp.272-279, 2011.
[5] M.Kassab, G.El-Boussaidi and H.Mili, *"A Quantitative Evaluation of the Impact of Architectural Patterns on Quality Requirement"*, Springers Studies in Computational Intelligence Book Series, vol.377, pp.173-184, 2011.
[6] C.T.Su and D.Yeh, "Software Architecture Recovery and Redocumentation Tool of a Hospital Information System", in *International Conference on Computer and Communication Engineering(ICCCE'12)*, KualaLumpur, pp.143-146, 2012.
[7] F.Buschmann, R.Meunier, H.Rohnert, P.Sommerlad and M.Stal, *"Pattern Oriented Software Architecture"*, Wiley India Pvt Ltd, New Delhi, 2010.
[8] M.Galster, A.Eberlein and M.Moussavi, "Systematic Selection of Software Architecture Styles", *IET Software*, vol.4, no.5, pp.349-360, October 2010.
[9] S. Srivastava, *"Archaware2: A Style based Software Architecture Documentation Tool"*,M.Tech Dissertation, Indian Institute of Technology, Kanpur, 2011.
[10] U. Banodha and K.Saxena, "Impact of Piper and Filter Style on Medical Process Reengineering", *International Journal of Engineering Sciences*, vol.4, pp.398-409, 2011.
[11] Y. Dajsuren and M.v.d.Brand, "Architectural and Design Patterns in Multimedia Streaming Software", *Proceedings of the third International Workshop on Software Patterns and Quality SPAQu09*, Tokyo, 2009.

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

[12] E. B. Fernandez and J.L.O. Arjona, "The Secure Pipes and Filters Pattern", *in 20th International Workshop on Database and Expert Systems Application DEXA'09*, pp. 181-185, 2009.
[13] I.Kazuyuki, I.Takashi, M.Keiichi, I.Mayuko, H.Shin, C.Yuuki and K.Yukio, "*Development of New Image Processing Framework by the Collaboration of FujiFilm and FujiXerox"*, FujiFilm Research and Development, vol.55, 2010.
[14] T.Peng, L.Sun and H.Bao, "Design and Implementation of ATM Simulation System based on MVC Pattern", *in International Conference on Educational and Information Technology (ICEIT 2010)*, pp.328-331, 2010.
[15] McGruder and C.A, "*MVC for Content Management on the Cloud"*, MS Thesis, Naval Postgraduate School, 2011.
[16] S. Fan, Y. Zhang and L. Zhang, "Study and Application of a Multimedia Content Transformation Method Based on Model View Controller 2x Pattern," in *Proceedings of the7th World Congress on Intelligent Control and Automation*, China, pp.6655-6658, 2008.
[17] C. Chen and H. Han, "The Research on Modern Distance Education System based on Improved MVC Pattern," in *International Conference on Computer Science and Software Engineering*, pp.462-465, 2008.
[18] X. Hou, N. Li, Y. A. Tian and H. Yang, "A Framework based on MVC of Document Processing," *in Proceedings of the 4th International Conference on Cooperation and Promotion of Information Resources in Science and Technology*, pp.290-294, 2009.
[19] M. Nowak and C. Pautasso, "*The Design Space for Modern HTML5/JavaScript Web Applications"*, Saturn Web Technologies, Minneapolis, 2013.
[20] C. Liu and K. Wang, "An Online Examination System based on UML Modeling and MVC Design Pattern," in *International Conference on Control Engineering and Communication Technology*, pp. 815-817, 2012.
[21] A.Holzinger, K. H. Struggl and M. Debevc, "Applying MVC in Design and Development of Information Systems," *Proceedings of the International Conference on e-Business (ICEB)*, pp.1-6, 2010.
[22] T. Iulia, M. Ciocarlie and H. Ciocarlie, "Best Practises in iPhone Programming," *in Proceedings of the International Conference on Computer as a Tool (EUROCON)*, pp.1-4,2011.
[23] Z. Liu, T. Li and G. Yang, "An MVC based Collaborative Framework Support for Test Suite," in *Proceedings of the 14th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp.172-177, 2010.
[24] N. Medvidovic, M. Mikic-Rakic, N. R.Mehta and S. Malek, "Software Architectural Support for Handheld Computing," *IEEE Computer*, vol 36, no. 9, pp. 66-73, 2003.
[25] A. Rao, S. Nandy, H. Nikolov and E. F.Deprettere, "USHA : Unified Software and Hardware Architecture for Video Decoding," in *9th IEEE Symposium on Application Specific Processors (SASP)*, pp.30-37, 2011.
[26] K. Cheng, B. Itzstein, P. Sztajer and M. Rittenbruch, *"A Unified Multi- touch and Multi-pointer software architecture for supporting collocated work on the desktop"*, Technical Report ATP-2247, NICTA, Sydney, 2009.

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)