Design and Implementation of Combinational Software Architecture for Batch Image Processing System

Mili Mary Philip \(^1\) and Vijayakumar B \(^2\)

Department of Computer Science, Dubai Campus
Birla Institute of Technology and Science, Pilani
\(^1\)p2A012003@dubai.bits-pilani.ac.in, \(^2\)vijay@dubai.bits-pilani.ac.in

Abstract

A software application is required to be built over a strong architecture and design. The failure to have a solid foundation would result in an application that is unstable and that is unable to support the changing business requirements. In this paper, we deal with an image processing application where images are to be processed using a combination of batch mode and interactive mode. The paper also gives a clear information on the factors that led to the selection of the pipes and filters architectural pattern for the data stream operations and the model view controller for the interactive part of the system. The combinational architectural design involving these two architectural styles has been focused in this paper. Flexibility is incorporated to the system by recombining the existing features, thereby making the application system open to enhancements. The user enters the image processing features and their order of processing as per the requirement. The images are processed in streams of input data and the resultant processed images can then be viewed by the user. The performance analysis involves measurement of the execution time for different test scenarios. This architecture can be deployed for many real-time application systems where both data stream and interactive mode of operations coexist.

Keywords: Architectural Pattern, Image Processing, Pipes and filters, Model View Controller, Software Architecture

1. Introduction

Batch image processing involves the processing of a group of images at a single shot. These images should be processed automatically without any user intervention. Such an application would be huge in size and highly complex as it involves many image processing algorithms for the processing of the complete set of images. This will require a careful software architectural design for the smooth and error-free working of the application system. 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 patterns play a very important role in building a software architecture with a defined property [5].

The architectural patterns are used to denote the structural properties of an application. They tackle a particular problem that repeats again and again in the design and development of an application system. This is performed by providing a solution for the problem. The solution consists of mainly the different components to be present in the system, the responsibilities of various components and the way they are to be connected. Thus, these patterns would have an impact on the architecture of the various components that makeup the application. Hence, the process of choosing an architectural pattern for an application is the most important and fundamental design criteria in the process of
software development. The success of the pattern is dependent on the fact that whether it meets the development and maintenance of the complete application system [5].

The present work deals with the development of an image processing application system, where both data stream and interactive modes of operation are involved. The architectural pattern chosen for such an application as well as the factors that led to its selection are also pointed down in the subsequent section. The analysis on the time taken to perform each of the image manipulations is also being recorded and compared with the time taken when the images are being manipulated in combinations.

2. Related Works

The section below deals with an overview of the existing works carried out in software architecture with a focus on the pipes and filters architectural pattern (P&F) and the model view controller (MVC) pattern. The factors that led to the selection of the above architectural patterns for the application under consideration have also been discussed here.

For the development of a large and complex software system, its architecture plays a very important role. Both the functional as well as the quality requirements are met with its software architecture [3-4]. Software architect plays a vital role in the software development process. It is the duty of the software architect to overcome the challenges in designing the architecture of a system with changing requirements [6]. The role played by a software architect stretches from analysis of the stakeholder requirements to the evolution of the application system [7]. An architecture of a system can be classified as the most ideal one, if the system satisfies all the quality attributes like reliability, performance, scalability and so on [8]. The components which are the computational elements and the connectors that make up the interaction among the components and their configurations together forms the architecture of a software system [9]. The process of making the choice of architectural pattern is the most important step in the software development process [5]. The architecture of a system is also considered as a bridge between the requirements and its implementation. In the present-day world, it is very much essential to understand the high-level relationship among systems, so that the new systems can be designed using existing reusable components and adding new modules.

There are a huge number of applications that process or transform a stream of data [10]. The entire application is divided into different stages. The processing of a batch of images, which is the major focus in this paper, is always highly complex and time consuming and a cumbersome task. Hence, separating the entire application into different processing steps is the best feasible approach. The architectural pattern that divides the complete system into different sequential processing steps is the P&F pattern. These steps are connected by the dataflow through the system [11]. The entire application consists of various image processing filters (components) which will manipulate the data and connectors (pipes) that takes up the data from one component to the next component for the processing [12]. The filters that make up the complete application carries out their tasks independently. Each processing module is unaware of anything that has happened before it received its data. These filters are connected by the pipes which decide the sequence in which the processing should take place. Pipes provide the interface to provide the input data as well as to send the output data to the next stage.

The multimedia framework [13] is designed with the P&F architectural pattern. The framework builds the streaming graph which in turn consists of nodes that would be the filters which will perform the processing of the application and the edges that act as the pipes to connect the different media components. This pattern has also been used in the field of medical process reengineering [14]. The security feature was also incorporated to different processing stages in the work by Eduardo B Fernandez [10]. The chances of reusability and flexibility is also increased by allowing different combinations of the
processing modules in the P&F pattern [15]. As discussed by the authors in their work [16], this pattern is mainly used in all applications where large streams of data are to be processed. A concurrency aware P&F framework is proposed in this work, to handle concurrent processing of streams of data. This architectural style is mainly used in applications where data is processed incrementally. Performance and flexibility are the two main advantages of this pattern that has been studied in the work [17]. The P&F architectural pattern can also be modelled for system of systems architecture [18]. The P&F pattern are used in systems where there is a flow of data and also when there is a change of components and connectors at the time of execution of the application.

The image processing application, discussed in this paper consists of a user interface that is to be used for capturing the user inputs so that the usability of the application can be enhanced [11]. Since the user interface is prone to changes and adaptation, it should be kept independent of the functional core. Thus, the architecture of the system should support the adaptation of user interface without creating any change in the functionality of the system. The MVC pattern is the best architectural pattern that supports this requirement and hence chosen for the interactive part of the application system. The interactive part of the application system is divided into three components namely – the model component, which consists of the set of images and the required functions using which the processing of images can be performed, the user interface where the user can input his requirements and the controller component which will handle the input given by the user.

The MVC architectural pattern has been used in the development of games [19]. In the development of the software for any game, it is quite essential that the game logic code is kept separate from the user interface and hence it is easy for transitions to different platforms. The authors in [20], suggests that the applications that are developed using the MVC style, are more extensible and reusable than others. The MVC architecture was considered as a good foundation for explaining and building an application on the iOS platform. The framework developed for human movement analysis [21], consists of an interface that is designed using the MVC pattern. The present-day systems require a high degree of user interaction which is made possible with the help of graphical user interfaces. The MVC pattern provides the best architectural organization for interactivity in software systems.

Few of the existing works using the P&F architectural pattern as well as MVC pattern, and the factors that led to the selection of these patterns have been pointed out here. These patterns are being used for the development of the software architectural design of an image processing system in this paper. The user interactivity as well as the stream processing of the images has been accomplished using the proposed software architectural design.

3. Architectural Design Description

This section describes the application system design considerations pertaining to the image processing system. An image database is created and stored at a specified location. The user gives the requirement for processing the images which include the following operations – image crop, resize, add border, rotate and brightness contrast. The above requirements are given as inputs through the user interface. The combinational architectural model [22] provides an environment for carrying out the above operations and finally the resultant images will be stored in a particular location which can be viewed by the user. The software architectural model for the image processing application is being detailed below. It includes interactive as well as batch processing mode of operations. Figure1 shows the architectural design for the image processing application.
Here, the entire functionality of the system is separated from the user interface. The user interface is prone to changes and hence should be made adaptable. The changes made in the user interface will not affect the core functionality. The current approach offers flexibility by separating the user interface from the functionality of the application. The interactive part of the image processing application is handled using the MVC approach. It is divided into three components - model, view and controller. Each of these components serve different purposes. The information is displayed to the user through the user interface which plays the role of the view component in this application. The user requirements, like the required image processing modules and their order of processing are entered in the user interface.

The add and order options for this purpose are shown in the Requirements component as shown in Figure 1. The processing modules that are not needed can also be deleted using the delete option. This will be handled by the controller which will in turn call the required modules present in the coreFunction of the model component. The model also consists of the set of input images (coreData) on which the processing has to be performed. The processed images will be saved to a particular location which can then be viewed by the user.

The image processing application system must process the images in streams of input data which is made possible by using the P&F architectural pattern. This system can be developed using multiple filter components which can handle the various processing functionalities. The various filter components are denoted as $F_1, \ldots, F_n$ in the Figure 1 above. The connectivity between the filters is made possible using pipes. The processed images from the first filter will be taken up by the pipe and will give it as input to the next filter. Flexibility of the system is a very important attribute that have to be incorporated to the system. The application system should be open to enhancements which can be made possible by recombining the existing processing modules or filters.
4. Implementation Details

The image processing modules that have been considered here for the different test scenarios are – resize, crop, rotate, add border and brightness contrast. The different modules as well as the order of its execution are user inputs as mentioned above. In Figure 1, the filters $F_1 \rightarrow F_n$ act as the filters and the connectivity between them is using the pipes.

The different steps for the execution of the application system is as shown below:

Step 1: Create an image directory and store all the input images in it.

Step 2: Choose the required image processing modules through the user interface. It can be either a single processing module or a group of modules.

Step 3: If more than one processing module is selected, then enter the order or the preferences in the way in which the processing has to be performed.

Step 4: Process the images using the above selected image manipulation filters.

Step 5: Save the processed images to the target directory.

Step 6: Measure the time taken for the execution of each of the modules.

The ImageMagick software [23] has been used to do all the image processing functionalities in this application. It supports and accepts the input images in different formats. It can either be used through the command line interface or otherwise through the various programming languages. JMagick is the object-oriented Java interface to ImageMagick.

5. Results and Discussion

The image processing application system has been developed and tested on a Linux server with UBUNTU 14.04 operating system. The implementation has been carried out using the programming language JAVA. A sample set of 500 google map images were taken for each of the test scenarios. The images were of different sizes ranging from 10kB-500kB. The total size of all the images was approximately 500 MB.

Test Case 1: In this scenario, the filters are considered separately. The time taken for each of the filter to work independently on the entire set of 500 images was recorded. It was seen that the processing of resizing the images took the maximum time when compared with the filters like crop, rotate, add border and brightness contrast. It takes 21.9 seconds for all the 500 images to be resized. The execution time for the various image processing modules are listed in the decreasing order as follows: crop (10.3seconds), rotate (9.5seconds), add border (9.2seconds) and the brightness contrast (8.5seconds). Figure 2 clearly depicts the differences in execution time for each of the image processing operations. Based on the analysis, as the resize operation took the maximum time for its processing, we have chosen the same for the subsequent execution scenario.
Test Case 2: As discussed in the previous section, the resize image processing option is chosen for the analysis in this scenario. The time taken for the image resizing operation to be performed on a set of 100 images were initially performed. It was observed that with each increase in the number of images, the execution time also shows an increase. Figure 3 depicts the relationship between the execution time and the number of images chosen. For the set of 100 images, it took close to 5 seconds to be resized. As the number of input images doubled, the execution time also showed a linear increase. The time taken for each set of input images is plotted in Figure 3. It can be inferred that there is a linear relationship between the number of images and its execution time.

Test Case 3: This scenario clearly shows the difference in the execution time for combinational mode of operation. The selected image processing features were to rotate, resize and to add the borders for the images. The execution time for each of these processes when performed individually were recorded as the first step and then plotted in Figure 4 below. It took 9.43 seconds for all the images to be rotated. The add border option also took almost the same time of 9.26 seconds while the resize operation took the
maximum time of 21.95 seconds. Thus, the total time taken for the rotate and add border option is close to 18.6 seconds. Similarly, the total time taken for the three operations will be approximately 40.64 seconds.

The images were then processed in a combinational mode. It was observed that it took only 10.55 seconds to complete the processing when the rotate and add border options were given together in combinational mode of operation. Thus, it can be inferred that the execution time shows a huge decrease when the image manipulations are executed in combinations of filters. When the resize, rotate and add border operations were given chosen together in a single iteration, it took close to 27.52 seconds for all the 500 images to be processed. The execution time is approximately reduced to half of the total time taken when these operations are performed separately. The graph in Figure 4, clearly shows the time taken for each operation when performed separately as well as in combination.

![Figure 4. Performance Graph for Combination of Filters](image)

Based on the experimental analysis, it can be inferred that the application would work effectively when the images manipulations are being performed in a combinational mode of operation. The graphs shown above clearly depicts the time taken for each of the filters for its execution. The combination of software architectural pattern that we had chosen for the above application system led to the development of an efficient system which could perform the processing of the images in streams and thereby creating a substantial decrease in its execution time. The user has the privilege to select all possible combination of filters through the user interface, according to which the images would be processed. The order in which the images should be processed is also decided by the user. This software architectural design supports the accomplishment of the filters in any order as per the user requirement. Thus, the software architectural design that is proposed in this work, provides the flexibility to use any combination of the available processing filters according to the demands of the user.

6. Conclusion

The software architectural design for the image processing system presented in this paper, supports data stream as well as interactive mode of operation. The user interface is developed so that the users can add new filters or delete any existing filter according to the requirement. This user interface also provides the flexibility to change the order of
processing of the images. As it is separated from the other components of the application system, any modification to the user interface, will not affect the working of the application system. There has been a considerable decrease in the execution time when the images were processed using the different combination of filters when compared with the time taken when each filter works independently. The filters perform the transformation on the input images successively until the complete set of images are processed.

The present work can be extended by constructing the traceability table and creating the architectural specification file. The traceability table can store information on the dependencies between the various image processing features such as resize, rotate, crop etc. The architectural specification file contains the details of various components and connectors used in the application system. Currently, we are working on the reflection architectural pattern which will make the application system more adaptable by accommodating the requirement changes at runtime.

Acknowledgements
The authors would like to express their sincere gratitude to Ms. Anubha V, Ms Celeste Sakhile and Ms. Raunak Gedam for their support in the experimental work.

References


