CSE – 681 BUILD SERVER

Instructor: Jim Fawcett

Project#4 Build Server- OCD

Version 2.0

Dwivedi Nilesh (SUID#751870908)

Date: 12/06/2017

Table of Contents

1 Executive Summary

In the process of software development, multiple people are involved from different locations. Each person can be doing same or different part of the same project. Thing are easy when the software product is small and relatively small group of people are working on the project but as the size and people associated with the software product increases, the management of different versions of software and to get updated from the changes are done in the project will become difficult. So, it is necessary to use project management tool. The process of building and testing the code should run parallel to utilize time of the team. Build server is the tool that build the source code when user request to do. It will build the code which will notify user about the build errors and warnings and the result of the build. Also, there can be problem of running the same code on different machine. So it can be possibility that the code will not run in some machine this can be resolved by build server, as the code is going to build on the build server only. So different users can see the results of the build instead of their local machine. Which can resolve "Running on my machine" problem. Because build server provides known and consistent environment and avoid the dependency on the local machine.

The purpose of the project is to accomplish the tasks and functionalities which are:

1. Build the code for different languages sent by the repository
2. Create and send the build log to track the build process
3. Prepare the test libraries for test harness where tests are running

The users of the project can be developers, testers, Managers, QAs, Teaching assistant and instructor. The developer will use this document and instruction of TA and instructor to meet the requirement of the project and address the current issue.

The risks described by the document of the project is:

- Long Builds
- Maintaining large volume of builds
- concurrent access
- Security
- Cost
- Usability
- Flexibility

This document will provide the possible solution of the critical issue and the risk also. It will also describe the package diagram which is interaction between programming modules. It will describe activity diagram which will be the flow of activities to be done by the build server.

2 Introduction

Project Management tools changed have been changed dramatically. It is now also focusing on different user groups instead of making it continent to developers only, it also focusses on managers, testers and QAs. Build server is module that will build the code sent by repository and prepare test libraries for test harness. Build server plays huge role in continuous integration process. Another advantage of build server is that the build that is going to tester/QA is directly from the repository which has the latest version of the code. By this way the project is trackable according to version. Also, it is considered as a good practice that we should everything the developer codes. Build server will build the sources from the repository and let the team know about where code is broken.

Build server is great tool for QA/Tester. The process of testing is made automated. They can keep track about the testcases and Also builds are coming from the latest version so the latest changes are also tested.

Build server is very useful tool for the distributed development where codes in the repository is committed to repository from all over the world and in different locations people also do not communicate face to face. For example, Open Source, R&D projects. On these situations where version control is heart of the system, the build server is useful product. So instead of making developer to worry about merging the project or risk of committing the bad build to the repository the testing process can be automated, the dependency can be checked. So, if developer commit something bad than it can be notified quickly so it decreases the worry of developer to think about the building and testing the new version.

2.1 Application Obligation

The primary goal of the system is to provide:

- To build the code sent by the Repository
- Parse the test request and select correct tool set to build the code
- Create the build log and send this log to Repository
- After build Success send test library along with TestRequest to Test Harness
- Send Request to Test Harness to execute test

2.2 Organizing Principle

The Build Server should communicate with repository and test harness. It will get input from repository as source code and test request as XML file, will build the source and make the testdriver executable and gives the output as builded code and TestRequest as XML file and send it to the test harness.

2.3 Objective and Key Idea

The Objective of the system are to perform primary functionalities which is to build the source and do necessary things to prepare test harness to run the tests.

The key idea behind the system is that by build server, everything that developer codes can be tested in a known and consistent environment which eliminate dependency on the local machine by this way it will release responsibility on developer to build and make It easy for tester to test project continually.

3 Users and Use Cases

3.1 Developer

The developer will understand the requirement and build the project accordingly. The developer will code for this system and implement all the functionality that system requires. Also, from this system the developer will implement Message-passing Communication Channel using WCF (windows communication foundation). Also, GUI will be built using WPF (windows presentation foundation) in project 3. And whole integration server with all the functionalities in project 4.

Also, the developer is going to commit code to the repository through command line interface.

3.2 Instructor/ TA

Instructor/TA will use this application to check weather all requirement is fulfilled or not. TA will also analyze facility for future project and also ensure that critical issues are correctly addressed. Other Applications.

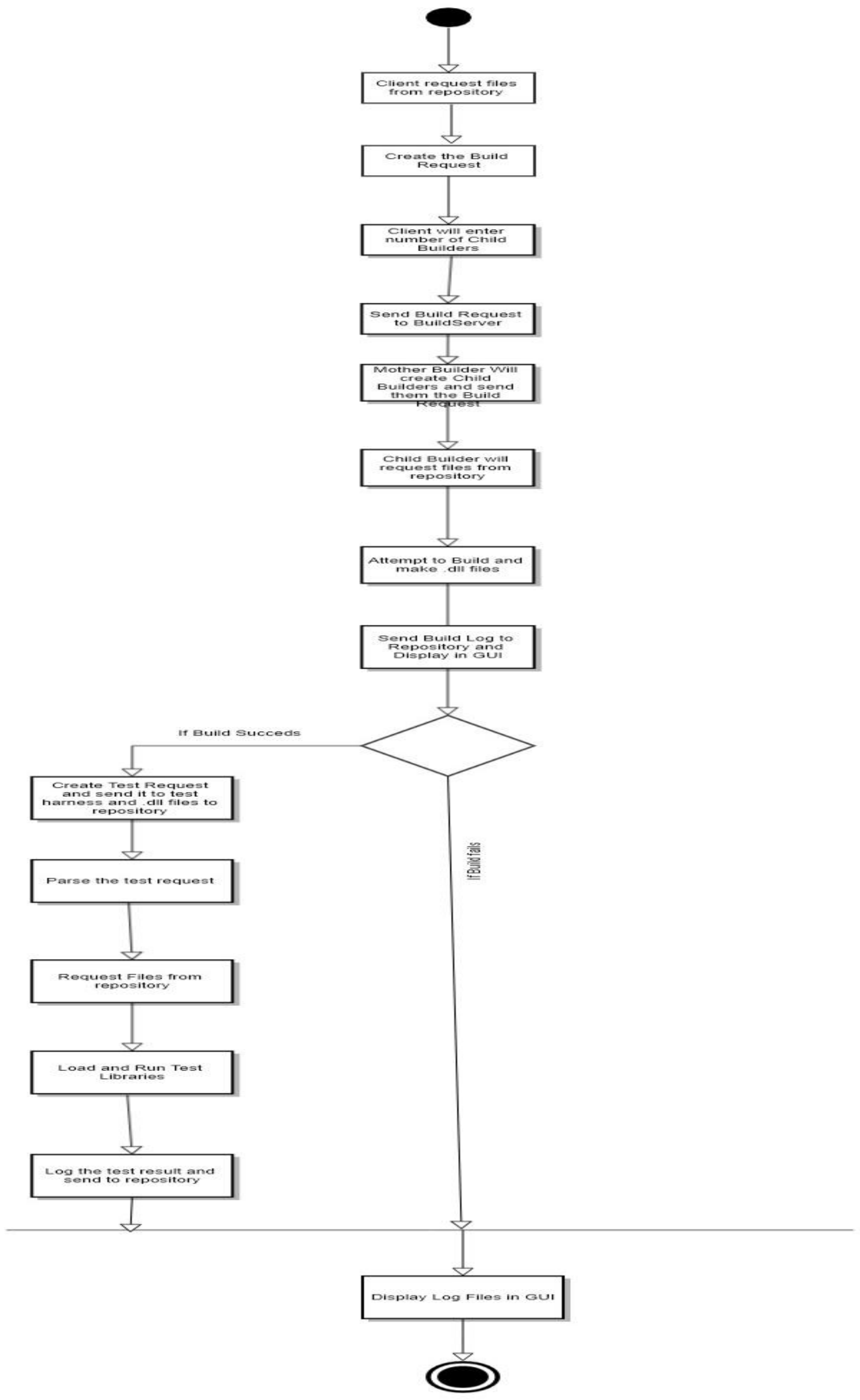
3.3 Tester

Tester is the user which is responsible for test the latest build. The tester need to authorized before accessing repository or build server. After the authorization, tester will test the build using pre-defined test cases and also holds the output. Tester will request for test and then test harness will run the test and compare the expected and real output. Tester will communicate with the system by well-defined GUI.

3.4 Project Manager

Project manager is the user who will manage the development of whole project and communicate with people associated with project for e.g. client, developer, tester, QA analyst and track the progress. Project manager can track the development by GUI provided by this application through the information contained by build log. Also, Project Manager can track progress of testing by test log in the repository send by test harness. The project Manager will communicate with the system with GUI that provide the information in the log files in a way that it can be more informative and understandable.

4 Activity Diagram


Client request files from repository
Create the Build Request
Client will enter number of Child Builders
Send Build Request to BuildServer
Mother Builder Will create Child Builders and send them the Build Request
Child Builder will request files from repository
Attempt to Build and make .dll files
Send Build Log to Repository and Display in GUI
If Build Succeds
If Build fails
Create Test Request and send it to test harness and .dll files to repository
Parse the test request
Request Files from repository
Load and Run Test Libraries
Log the test result and send to repository
Display Log Files in GUI

4.1 Description

(1) First, the client request for files from the repository

(2) The build Request is created according to the files selected by client.

(3) Client will enter the number of child process to be created.

(4) Repository will send build request to the build server.

(5) Mother Builder will create child builders and send build request to them.

(6) After Parsing Build Request, Child builder will request files from the repository.

(7) Child Builder will attempt to build the files according to the test request and make the .dll files

(8) Child Builder will send the build logs to the repository which will displayed by GUI.

(9) If all builds are succeeded then It will create the test request and send the test request to the Test Harness and .dll files to repository.

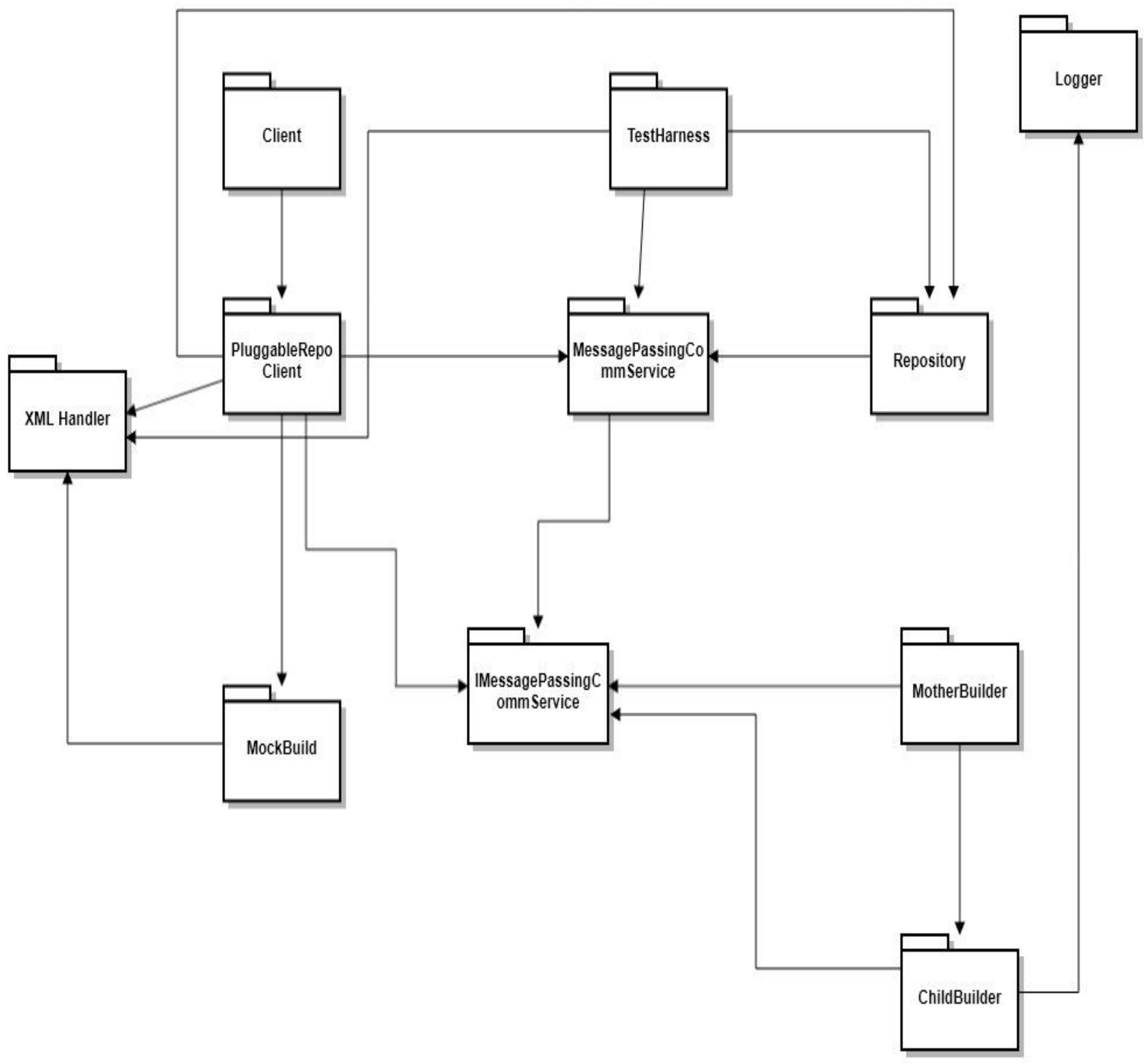
(10) Test harness will parse the test request and Request appropriate dll files to repository.

(11) Test harness will load and run the dll files and test them.

(12) It will send the logs to the repository.

(13) Repository will display the logs to GUI.

5 Package Diagram



5.1 Repository
The repository package will contain functionalities of repository server. It will send the build requests to mother builder. Receive the file request from other components using WCF (Message Passing comm Service package) and send the desired file to specified location from the RepoStorage.

5.2 Mother Builder
This package will receive the build request from the repository and create the child processes (Process pool). It will assign the build request to child builder as they are available.

5.3 Child Builder
Child Builder package is the core of the build server. It will receive the build request from the mother builder. It will parse the build request and request for files from repository. After receiving files from repository, it will use functionalities of MockBuild to build the files. If build is successful then it will create test request, sent .dll and log files to Repository and Test request to Test Harness. If build fail then it will send only logs to repository.

5.4 Mock Build
Mock Build package will build the list of files and make dll files from it. It will generate exception if build fail. This package will also communicate with XML handler and create TestRequest.

5.5 MockTestHarness
This package will provide a mechanism for loading dll files and running test on that. It will receive testrequest from child builder and parse that request. It will request dll files from repository. And load that dll files and log the result. After that it send log files to the repository.

5.6 Logger
Logger package is used for logging the message on the file. It provides the way to write on the file. All package will log the messages to the log file which can be later showed on GUI.

5.7 PluggableRepoClient
The PluggableRepoClient package will provide Graphical User Interface to the system. The Client can interact with the PluggableRepoClient to browse the files from repository and create a build request.

Moreover, the build logs and the test logs will also be available on the Graphical User Interface for the client to view.

5.8 MessagePassingCommService

The task of MessagePassingCommService Package is to communicate with all of the major packages in the Build Server. The MessagePassingCommService package provides message passing mechanism to communicate among each other. It is also responsible for transferring files from one location to another.

5.9 IMessagePassingCommService

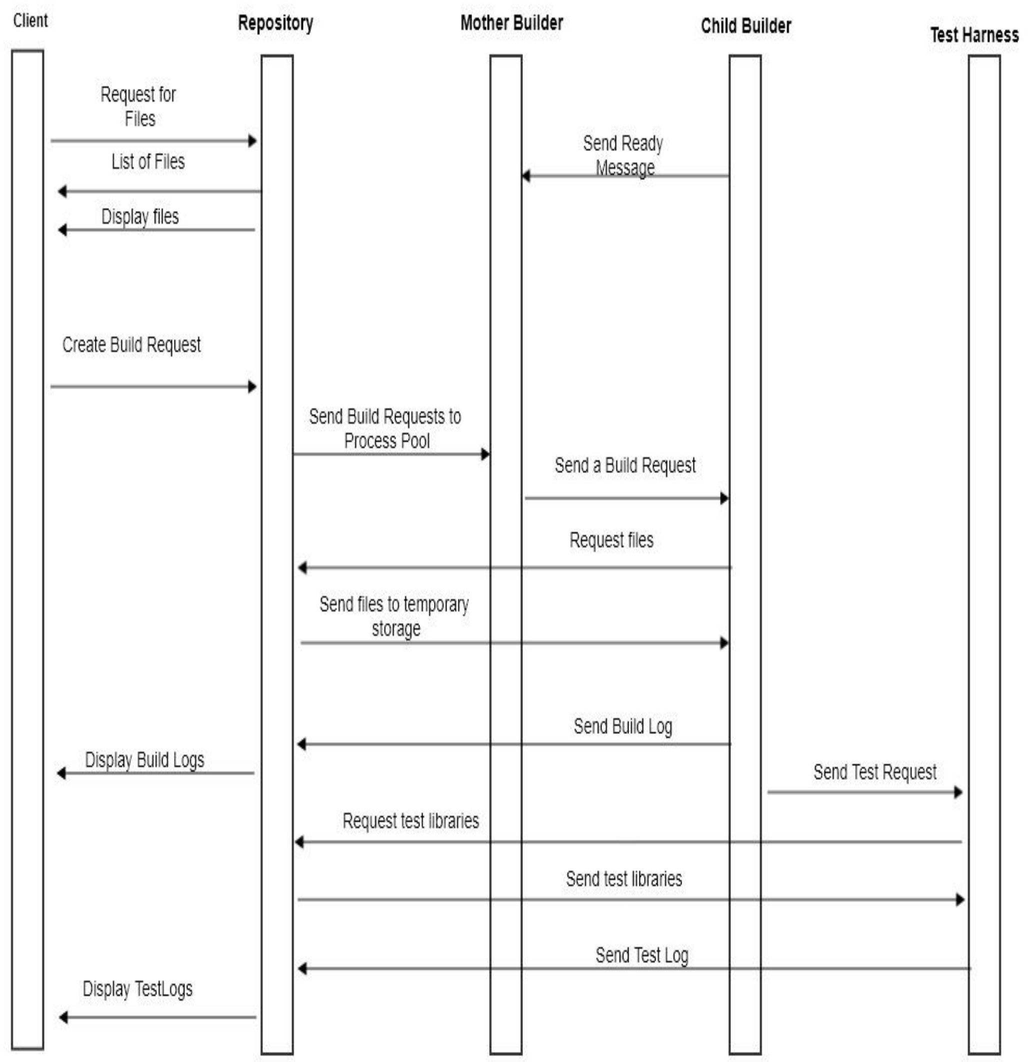
IMessagePassingCommService package provides service interface for MessagePassingCommService. This package provides an interface for message passing and file transfer.

The class CommMessage represents serializable messages which is sent between all packages for communication between them.

5.10 XML Handler

This package contains basic functionalities of XML handling like parsing the XML, Making new XML file. The service this class will be used by ChildBuilder, TestHarness and PluggableRepoClient.

6 Message Flow



6.1 Client

1. It requests for files from the repository
2. It creates the build request and send it to the Repository

6.2 Repository

1. It sends the list of files to the Client
2. It will send files like BuildLog, TestLog to the client so that it can be displayed on GUI.
3. It will send build request to the process pool.
4. It will send files to the storage like buildstorage and test storage when the files is requested.

6.3 Mother Builder

1. It will assign send build request to one of the child builder.

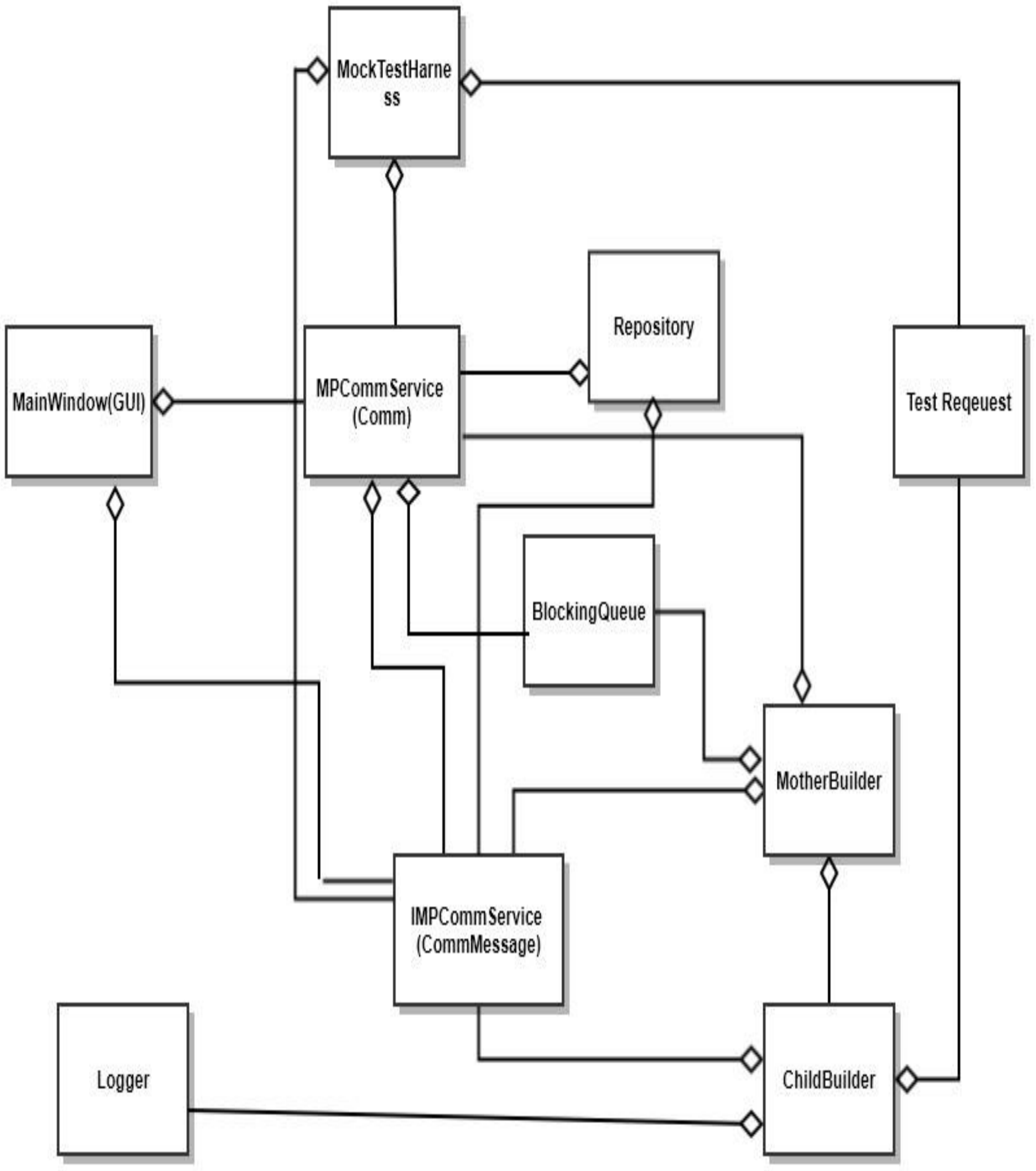
6.4 Child Builder

1. It will send ready message to mother builder which indicate child to be in ideal condition.
2. It will request for files in repository.
3. It will send test request to the test harness.
4. It will send build logs to the repository which later displayed on GUI.

6.5 Test Harness

1. It will request for files from the repository.
2. It will send the test log to the repository.

7 Class Diagram



7.1 BlockingQueue

The major task of Blocking Queue is adding the Build request from the repository in a queue and which is later accessed by mother builder. If a thread request to pop from blocking queue then the queue will block the thread and wake up when any insert operation on queue is performed.

The BlockingQueue class has an aggregation relationship with MPCommService class.

7.2 MockBuild

The MockBuild class create and parse XML file and build the required file and make .dll file.

The MockBuild class has an aggregation relationship with MainWindow(GUI) Class. In MainWindow (GUI), we create object of BuildRequest class to create build requests.

7.3 MotherBuilder

The MotherBuilder class processes the process pool which contains all the build request. The BuildServer maintains a record of all the child builder who are ready to process the build request and whenever a build request arrives it sends to the child builder.

The MotherBuilder has an aggregation relationship with IMPCommService and BlockingQueue.

7.4 ChildBuilder

The ChildBuilder class process the build request and Log the result into file then sent it to repository.

The ChildBuilder has an aggregation relationship with IMPCommService, MotherBuilder and Logger.

7.5 Repository

The Repository class will provide the functionalities to browse the files from repository and to create a build request and also sent it to mother builder.

The Repository class has an aggregation relationship with the MainWindow Class (Graphical User Interface).

7.6 DLLLoader

DLLLoader class will provide a functionality for loading dll files and also it will run the test function from the dll file and return the result which help to know test passed or fail.

The DLLLoader class has an aggregation relationship with TestHarness class. The TestHarness class creates an object of DLLLoader class to load and run the test library.

7.7 IMPCommService

The IMPCommService class contains the service interface for MPCommService. It provides an interface for message passing and file transfer.

The IMPCommService class has an aggregation relationship with MPCommService class.

7.8 Logger

The Logger class provides the functionalities to create a log file and write the specific message on file.

The Logger class has an aggregation relationship with BuildServer to create build log, TestHarness to create test logs.

7.9 MainWindow

The MainWindow class is a Graphical User Interface for Pluggable Repository. The Class communicates with repository. It also displays the build logs and test logs.

The MainWindow class has an aggregation relationship with Client class, Window_2 class and BuildRequest class.

7.10 MPCommService

The MPCommService class provides functionalities to communicate between two modules of the system.

The MPCommServiceclass has an aggregation relationship with TestHarness, Repository, MainWindow (Graphical User Interface) and Blocking Queue.

7.11 TestHarness

The TestHarness class provides the functionalities to load and run the test libraries. Also, it requests the files from repository after parsing the test request.

The TestHarness class has an aggregation relationship with DLLLoader Class.

7.12 Window_2

The Window_2 class provides the functionality to read the contents of the source code file, test driver file, build log files and test log files.

The Window_2 class has an aggregation relationship with MainWindow (Graphical User Interface) class.

8 Critical Issues

8.1 Long Builds

As component in the product is growing, the more lines of code and files should be maintained. And longer the code, the longer time it will take to build. This will affect the product development. In agile development where builds are expected to run frequently and organization is dependent on the output of the build then it will slow down the speed of development. It will also create issues like Long developer wait time, long queue, late notification for even small bug.

8.1.1 Solution

If the project size is big then only one server will not work and create this problem. Making more than one server OR server with good processing power do the job. When using multiple servers, the task can be divided among them so that builds will take less time.

8.2 Security

Security is the major concern while working with the software product releases. A system should need to confirm that the person has specific privilege or not to perform certain action. Secure system is needed to protect the confidentiality of the code as this system can be attacked by hackers to break/steal the code.

8.2.1 Solution

To solve this issue Authentication and authorization should be properly implemented. We can use programming language internal functionality for that. The system should properly check about the user information before giving access to files.

Also, the message between sender and receiver can be encrypted to increase the security which prevent unauthorized access.

Effective auditing and logging is the key to non-repudiation. A user cannot deny performing an operation or transaction.

Hashing algorithms are used to guarantee integrity of data such that it prevents loss of data during communication.

8.3 Cost

As the size of project increases, to prepare a smooth build server and avoid the long time taken to build, we should increase the capability of the server. So, the cost to run build server will increase. Also, storage of build server should be increase as the size of code or project increases.

8.3.1 Solution
Cost can be cut finding cheaper resource and shortening the project duration.

8.4 Usability
The usability of system depends upon how the system communicate with user effectively. GUI define a way to communicate with user. All the activity that user can perform have to have a way to do that form the GUI. All the user in the system will have a GUI to use the system. Effectively building GUI and increase usability is also an issue of this system like any other interactive system.

8.4.1 Solution
Windows presentation foundation provides an effective way to create an interactive GUI. Windows Presentation Foundation (or WPF) is a graphical subsystem by Microsoft which is used to build GUI. WPF will help developer the simulate activity or task and to provide user the way to do that.

To make searching faster the better indexing algorithm can be used for storing files.

8.5 Flexibility
As the source code which send from repository to build server can be in any language so This build server should provide the support else the project will fail to debug. Build server should have to support or provide the environment needed to build the code of that language.

8.5.1 Solution
Build server can add support for multiple languages and after analyzing the source code it will select the correct tool chain to build the code.

8.6 Large Testing Loads
The huge number of testing request at the test harness should be managed properly.

8.6.1 Solution
Multithreading can be a way out. We can run multiple thread concurrently, each server for one test request.

8.7 Repetitive Testing
Sometimes, Tester will pick files that is already tested. This is the case when there are many testers.

8.7.1 Solution
To prevent this, The tester will forced to write the details in test record before they attempt to test. Also, tester will need to look at test record once before attempt to test.

8.8 Presenting information to user
It is very important to present the information to the client in a useful manner such as client can easily check past test result and can take appropriate steps in order to continue future development cycle.

8.8.1 Solution
Client can check log files to keep track of the process and also after the test, Logs will be sent to repository for future reference of client.

8.9 Performance
Performance is the one of the critical issue which need to keep in mind when designing the system.

When Build server receive multiple test request then it will difficult to process it and it will affect the performance of the system.

8.9.1 Solution
Blocking Queue can be used to solve this problem. The incoming request can be stored in blocking queue and retrieved back when required. So sender does have to wait until Builder serve is not ideal.

8.10 Single Message Structure
How to generalize the message sent from one package to another package.

8.10.1 Solution
In the message to and from will hold sender and receiver address while body will contain the useful information to specific to the message.

9 Conclusion

In conclusion, the build server is a great tool for the management of software product with versioning and keep the team updated with the testing. This OCD explain the possible users of the system with the use case and Way of use (GUI/Command Line) of each one of the user. The usage and internal functionalities are defined using different diagrams.

I have covered all the activity being carried out by the build sever using diagram x in section y. This will describe how the activities are being carried in the build server in the sequence which is used to understand high level functionalities of build server. Also, I have discussed about the critical issue and risks about the project in section x. I have developed package diagram which help to develop the programming structure. So, it can be assured that through this OCD, the final software product can be built.

10 Comment on changes

In users section QA tester (Quality Assurance) is removed because tester and QA tester does the same thing.

In the Partition section, following packages are added to the system: PluggableRepoClient – For GUI, MessagePassingCommService – For communication and MotherBuilder – For spawning child builders.

Added a class diagram section and message flow diagram section which shows the flow of messages.

In the activities section, added new activities regarding GUI and message passing communication.

In the critical issues section, added few more critical issues.

11 Comment on deficiencies

The Build Server is only capable of Building the code in C# but not in other languages.

The poor authentication system is also major deficiency of the system as the security of the system is in danger.

12 References

https://ecs.syr.edu/faculty/fawcett/handouts/webpages/fawcettHome.htm

https://blog.codinghorror.com/the-build-server-your-projects-heart-monitor/

https://en.wikipedia.org/wiki/Distributed_development

https://stackoverflow.com/questions/1099133/what-is-the-point-of-a-build-server

https://stackoverflow.com/questions/2366135/why-should-my-development-team-have-a-build-server

https://stackoverflow.com/questions/1099133/what-is-the-point-of-a-build-server

https://electric-cloud.com/plugins/build-automation

https://www.infoworld.com/article/2980677/application-architecture/implement-a-simple-logger-in-c.html