**DEVELOPING PLUGINS FOR ALTAIR HYPERWORKS AUTOMATE**

**(A PROCESS AUTOMATION FRAMEWORK)**

*A Postgraduate Project Report submitted to Manipal University in partial fulfilment of the requirement for the award of the degree of*

**MASTER OF TECHNOLOGY**

**in**

**Computer Science & Engineering**

*Submitted by*

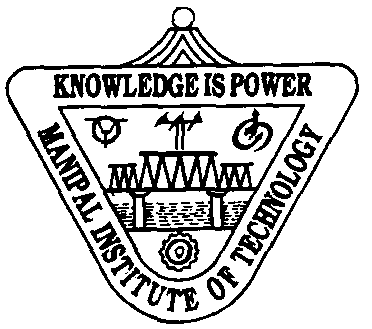
**Kavathiya Harshad Ramaniklal**

*Under the guidance of*

**MR. DEEPAK KUMAR DR. RENUKA A.**

**Senior Manager & Professor and HOD**

**Altair Engineering India Pvt. Ltd.**

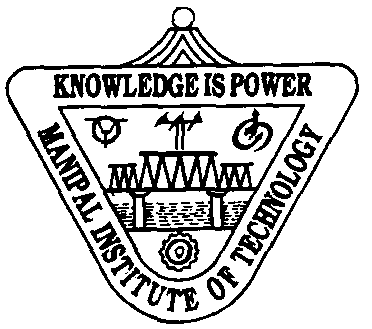
**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**MANIPAL INSTITUTE OF TECHNOLOGY**

(A Constituent Institute of Manipal University)

MANIPAL – 576104, KARNATAKA, INDIA

**May 2016**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**MANIPAL INSTITUTE OF TECHNOLOGY**

(A Constituent Institute of Manipal University)

MANIPAL – 576 104 (KARNATAKA), INDIA

Manipal

16-05-2016

**CERTIFICATE**

This is to certify that the project titled **DEVELOPING PLUGINS FOR ALTAIR HYPERWORKS AUTOMATE (A PROCESS AUTOMATION FRAMEWORK)** is a record of the bonafide work done by **KAVATHIYA HARSHAD RAMANIKLAL** (*Reg. No. 140913004*) submitted in partial fulfilment of the requirements for the award of the Degree of Master of Technology (M.Tech.) in **COMPUTER SCIENCE & ENGINEERING** of Manipal Institute of Technology Manipal, Karnataka, (A Constituent Institute of Manipal University), during the academic year 2015-16.

|  |  |
| --- | --- |
| **Internal Guide** | **Dr. Renuka A**  *HOD, CS & E Dept.*  *M.I.T, MANIPAL* |



**ACKNOWLEDGMENTS**

Apart from our own efforts, the accomplishment of any project depends to a great extent on the support and guidelines of numerous others. I take this chance to express my appreciation to the general population who have been instrumental in the effective finish of this Project.

I acknowledge and express my earnest thanks to beloved Director **Dr. Vinod V. Thomas**, Manipal Institute of Technology, Manipal who is a source of inspiration and who has been in the college to brighten the vision in technical education.

With profound respects and significant admiration, I take this chance to show my profound feeling of appreciation and obligation to **Dr. Renuka A.**, Professor and HOD, Computer Science & Engineering Department, Manipal Institute of Technology, Manipal for her rousing direction, helpful feedback and significant recommendation all through this project work. It would have been difficult for me to complete this project without her assistance and steady support. Furthermore, I am really thankful to her for providing this opportunity to me to deal with this thesis and permitting me the entrance to important resources in the department.

I wish to express my feeling of appreciation and obligation to my external guides **Mr. Deepak Kumar**, Senior Manager, Altair Engineering India Private Limited., Bangalore and **Mr. Narayan Rangarajan**, Program Manager, Altair Engineering India Private Limited, Bangalore for their inspiring guidance, encouragement, and untiring efforts throughout the course of this work. I also express my feeling of appreciation to the whole Automate Team, Altair Engineering India Pvt. Ltd., Bangalore for their help.

I am thankful to all my colleagues at Altair and friends from M.Tech. (CSE/CSIS) for their help during this present work.

Finally, I express my appreciation to the Faculty of Computer Science and Engineering Department, Manipal Institute of Technology, Manipal without whose support, this project would not have been a success.

**ABSTRACT**

HyperWorks Automate is a new generation process framework for CAE engineers that follows the flow based programming paradigm. This framework allows the developers to create reusable and modular CAE tasks using C++ or ITCL. These tasks are then shared with the CAE engineers as blocks which are basically black boxes with inputs and outputs. The CAE engineer can create a complex process visually by dragging, dropping and connecting the blocks on a process palette. The framework allows creation of processes easily without the knowledge of programming thus allowing a CAE engineer to focus on his domain.

This framework requires continuous enhancement to support the task developer needs. Some of them being

**Persistence of task in database:**This feature has become a necessity with the increase in number of tasks authored. This would allow for better task management via centralization and versioning.

**Test framework for Tasks:**  A test framework for the automated task would allow the developers to enhance the reliability and stability of tasks authored. This would allow mundane testing job to be automated.

**Inline debugger:**  Usability tests have suggested that developer’s productivity can be enhanced with the help of a task debugger. This feature is envisioned to help the developer to debug task during development.

In this project the above features have been conceptualized and a beta version of the code has been developed.

The scope of the work, technology and implementation methodology was well defined as per company standards, hence a waterfall model was adopted.

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table No** | **Table Title** | **Page No** |
| 3.1 | Methods defined for Inline Text Editor | 21 |
| 3.2 | Methods defined for BVT Infrastructure | 23 |

**LIST OF FIGURES**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Figure No** | | | **Figure Title** | | | **Page No** | |
| 1.1 | | | Company Product Structure | | | 1 | |
| 2.1 | | | HyperWorks Automate | | | 5 | |
| 2.2 | | | Task Block in HyperWorks Automate | | | 6 | |
| 2.3 | | | Model browser in HyperWorks Automate | | | 6 | |
| 2.4 | | | HyperWorks Collaboration Tools | | | 7 | |
| 2.5 | | | Procedure to share Tasks, Processes or Catalogues | | | 8 | |
| 2.6 | | | Debugging in HyperWorks Automate | | | 9 | |
| 2.7 | | | BVT execution flow | | | 10 | |
| 3.1 | | | Waterfall Model | | | 12 | |
| 3.2 | | | Database Persistence Module for HyperWorks Automate | | | 14 | |
| 3.3 | | | Content of Profile | | | 15 | |
| 3.4 | | | libraryprofiles.xml file content | | | 16 | |
| 3.5 | | | Main Folder Content | | | 16 | |
| 3.6 | | | Parsers.xml File Content | | | 17 | |
| 3.7 | | | Example of Automate Document | | | 17 | |
| 3.8 | | | Debugging Module for HyperWorks Automate | | | 20 | |
| 3.9 | | | BVT Infrastructure Module for HyperWorks Automate | | | 22 | |
| 3.10 | | | Snapshot of Perforce Client | | | 24 | |
| 4.1 | | | Snapshot of HyperWorks Automate Toolbar | | | 25 | |
| 4.2 | | | Snapshot of Export Catalogue Module | | | 26 | |
| 4.3 | | | Snapshot of Import Catalogue Module | | | 26 | |
| 4.4 | | | Snapshot of IMathTask loaded in Automate Canvas | | | 27 | |
| 4.5 | | | Snapshot of IMathTask code open in Debugger Browser | | | 28 | |
| 4.6 | | | Snapshot of Instant Suggestion given by Debugger Browser | | | 28 | |
| 4.7 | | | Snapshot of Goto Line Module in Debugger Browser | | | 29 | |
| 4.8 | | | Snapshot of Find String Module in Debugger Browser | | | 29 | |
| 4.9 | | | Snapshot of Find & Replace String Module in Debugger Browser | | | 29 | |
| 4.10 | | | Snapshot of old BVT Execution approach using command prompt | | | 30 | |
| 4.11 | | | Snapshot of Launch Automate Tests Browser | | | 30 | |
| 4.12 | | | Snapshot of BVT Browser in HyperWorks Automate | | | 31 | |
| 4.13 | | | Snapshot of Run BVT Option In BVT Browser | | | 32 | |
| 4.14 | | | Snapshot of Execution Status of BVT (Running) | | | 33 | |
| 4.15 | | | Snapshot of Execution Status of BVT (Success) | | | 34 | |
| 4.16 | | | Snapshot of Execution Status of BVT (Failure) | | | 35 | |
| 4.17 | | | Snapshot of BVT Log Report (Success) | | | 36 | |
| 4.18 | | | Snapshot of BVT Log Report (Failure) | | | 36 | |
| **Contents** | | | | | | | | |
|  | | | | | | | Page No | |
| Acknowledgement | | | | |  | | i | |
| Abstract | | | | |  | | ii | |
| List Of Tables | | | | |  | | iii | |
| List Of Figures | | | | |  | | iv | |
|  | | | | | | | | |
| **Chapter 1** | | | | **INTRODUCTION** | | | 1 | |
|  | **1.1** | Organizational Profile | | | | | 1 | |
|  | **1.2** | Motivation | | | | | 2 | |
|  | **1.3** | Objective | | | | | 3 | |
|  | **1.4** | Scope | | | | | 3 | |
|  | **1.5** | Target Specification | | | | | 3 | |
|  | **1.6** | Project Work Schedule | | | | | 3 | |
|  | **1.7** | Organization of Report | | | | | 4 | |
|  | | | | | | | | |
| **Chapter 2** | | | | BACKGROUND THEORY | | | 5 | |
|  | **2.1** | Introduction | | | | | 5 | |
|  | **2.2** | HyperWorks Automate | | | | | 5 | |
|  | **2.3** | HyperWorks Collaboration Tools | | | | | 7 | |
|  | **2.4** | Workflow of existing modules in HyperWorks Automate | | | | | 8 | |
|  | | | | | | | | |
| **Chapter 3** | | | | **METHODOLOGY** | | | 12 | |
|  | **3.1** | Waterfall Software Development Methodology | | | | | 12 | |
|  | **3.2** | Implementation Details | | | | | 14 | |
|  | **3.3** | Tools Used | | | | | 24 | |
|  | | | | | | | | |
| **Chapter 4** | | | | **RESULT ANALYSIS** | | | 25 | |
|  | **4.1** | Introduction | | | | | 25 | |
|  | **4.2** | Result analysis | | | | | 25 | |
|  | | | | | | | | |
| **Chapter 5** | | | | **CONCLUSION AND FUTURE SCOPE** | | | 37 | |
|  | **5.1** | Conclusion | | | | | 37 | |
|  | **5.2** | Future Scope of Work | | | | | 37 | |
|  | | | | | | | | |
| **REFERENCES** | | | | | | | 38 | |
| **PROJECT DETAILS** | | | | | | | 39 | |

**CHAPTER 1**

**INTRODUCTION**

The introduction includes a brief view of the company profile, details about the HyperWorks Automate. At the end of this chapter the project schedule details and organization of the project report (Chapter wise) are discussed.

**1.1 Organizational Profile**

* Altair is one of main worldwide supplier of innovation that reinforces customer advancement.
* Altair was founded by Mark Kistner, James R Scapa and George Christ in 1985.
* Altair engages customer development and basic leadership through innovation that upgrades the examination, administration and perception of business and building data.
* Altair’s products and services includes: HyperWorks, PBS Works Business Analytics, Solid Thinking, Toggled, Product Design, Thinklabs etc.
* Altair has more than 2,000 representatives worldwide and works more than 40 workplaces all through 22 nations.
* With a 30-year-in addition to reputation for top of the line programming and counselling administrations for designing, processing and enterprise analytics, Altair reliably conveys an upper hand to clients in a wide scope of commercial enterprises.

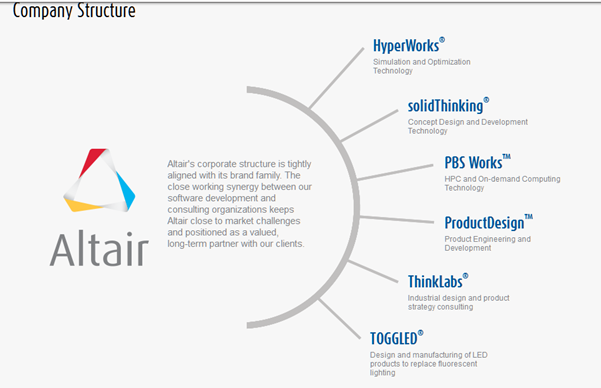


Fig. 1.1 Company Product Structure

* Altair provides services to almost 3,000 customers in various domain like government organisation, Aerospace, Architecture, Engineering, Construction, Automotive & Transportation, Consumer Product, Electronics, Energy, Heavy Industry, Life Science and Biomedical, Marine & Shipbuilding etc.
* Altair likewise has customer nearness in the gadgets, design building and development, and vitality markets.
* Globally, high demand of post-processors which defines the physical properties using user interface graphics (GUI) in computer aided engineering (CAE) has driven engineers to develop efficient and cost associated products in short time.
* Being very rare in the worldwide PLM software development community, transformation of Altair has taken place into a worldwide, multi-disciplinary item improvement consultancy consisting of 700 architects, designers, imaginative scholars and researchers.
* Altair’s strategic business focus on consulting to tackle the market challenges thrown by the industry and requirements of the clients needed on actively supporting software development strategy of Altair with complex problem solving and technology transfer partnering with their clients.
* At Altair, innovation is the main philosophy followed for working, which has been completely visible throughout all the project work.

**1.1.1 HyperWorks Automate**

HyperWorks Automate [4] is a new generation process framework for CAE engineers that follows the flow based programming paradigm. This framework allows the developers to create reusable and modular CAE tasks using C++ or ITCL. These tasks are then shared with the CAE engineers as blocks which are basically black boxes with inputs and outputs. The CAE engineer can create a complex process visually by dragging, dropping and connecting the blocks on a process palette. The framework allows creation of processes easily without the knowledge of programming thus allowing a CAE engineer to focuses on his domain.

**1.2 Motivation**

HyperWorks Automate framework requires continuous enhancements to support the task developer needs. Extension to the the existing feature of this product will motivate its existing users as well as prospect users to rely more on this product and also get more benefits with more convenience with time and resource saving

**1.3 Objective**

The objective of this project is to explore and add new features to HyperWorks Automate as follows:

* Integration of HyperWorks Automate with SQLite [7] database to ensure Database persistency
* Introduce Debugging Feature[2] in HyperWorks Automate
* Propose Build Verification Test (BVT)[1] Infrastructure in HyperWorks Automate

**1.4 Scope**

* Integration of the Automate with SQLite will give new capability to Automate to use and share catalogues among Team Members. Team Members can easily share their catalogues with each other. And if any modification done in catalogues it can easily reflect to all of the team member.
* Introduction of debugging feature in Automate will give capability of locating and fixing or bypassing bugs which makes Automate convenient and easy for people involved with it.
* By introducing BVT Infrastructure in HyperWorks Automate, it will give new capability to it to execute any BVT directly from HyperWorks Automate without manually writing lengthy and complicated code and also get rid of setting environment variable in system every time.

**1.5 Target Specification**

* By enhancing functionality, HyperWorks Automate becomes more user friendly and more and more users can rely on it to maximize performance and meanwhile save time and resources.
* Reduce development time with the help of BVT infrastructure.

**1.6 Project Work Schedule**

* Total duration: 10 months (August 2015 – May 2016)
* August 2015 to September 2015 – Understand the workflow of HyperWorks Automate. Understand the objectives and goals of the project.
* October 2015 to December 2015 – Integration of HyperWorks Automate with SQLite to make it Database Persistence. Understanding the requirement of Debugging feature for HyperWorks Automate.
* January 2016 to March 2016 – Implementation of debugging feature. And Understanding BVT creation and execution workflow.
* April 2016 to May 2016 – Implementation of BVT Infrastructure and bug fix.

**1.7 Organization of Report**

The report contains 5 well organized chapters. The 1st chapter describes the details about the organization profile, project, motivation, objectives and schedules of the project. Chapter 2 describes the background theory. Chapter 3 defines the methodology and implementation details. In chapter 4, end result analysis is done. The conclusion and future scope/enhancements are covered in the chapter 5.

**CHAPTER 2**

**BACKGROUND THEORY**

**2.1 Introduction**

The Background Theory includes a brief view of the HyperWorks Automate, HyperWorks Collaboration Tools and detail workflow of existing modules in HyperWorks Automate.

**2.2 HyperWorks Automate**

HyperWorks Automate is a new generation process framework for CAE engineers that follows the flow based programming paradigm [3]. Launch Automate from any HyperWorks Desktop client to build the process in three easy steps:

1. Drag and drop task blocks into the Process Window

2. Connect the tasks to define a workflow

3. Execute the process

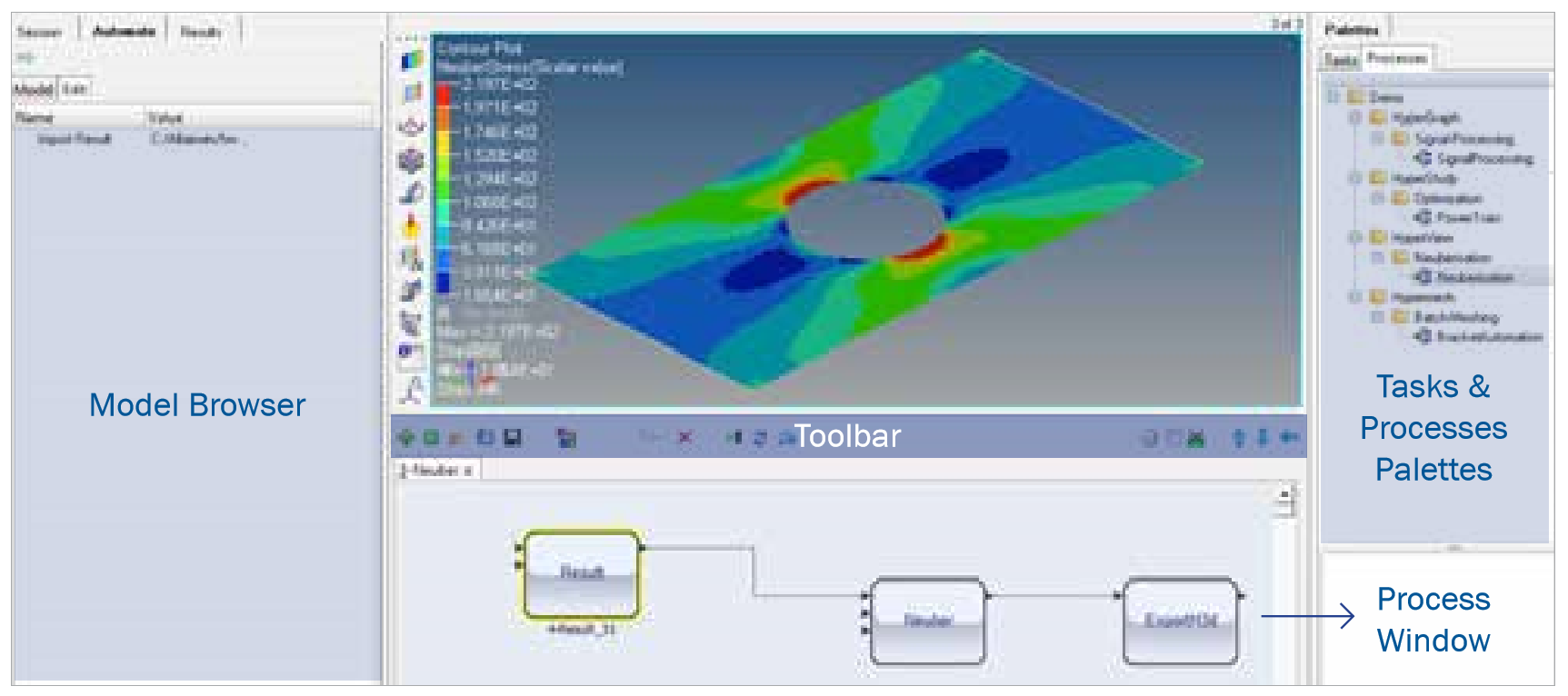


Fig 2.1 HyperWorks Automate

* **Process Window** helps to construct, debug, and run process diagrams (Refer Fig. 2.1)
* **Toolbar** allows users to perform key operations and execute a process (Refer Fig. 2.1)
* **Tasks blocks** contain all task information, which can be can easily move and assembled into process diagrams (Refer Fig. 2.2)
* **Model browser** displays current process components and properties (Refer Fig. 2.3)

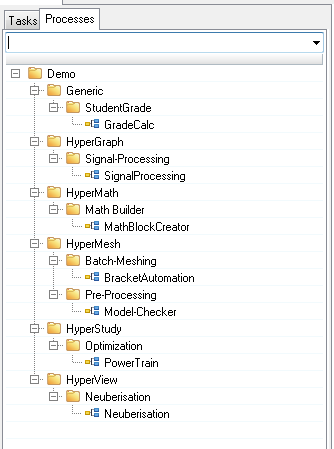


Fig 2.2 Task Block in HyperWorks Automate

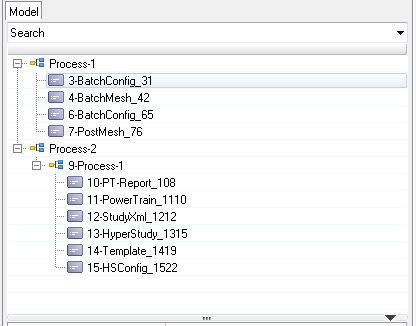


Fig 2.3 Model browser in HyperWorks Automate

**2.3 HyperWorks Collaboration Tools**

HyperWorks Collaboration Tools [5] is used during simulation life cycle. Team members used it for organizing and managing the files and data. The Collaboration Tools can be used out-of-the-box by every individual HyperWorks user, and it can be used by teams, shared over the network, enabling data access and collaboration on CAE projects. (Refer Fig. 3.4)

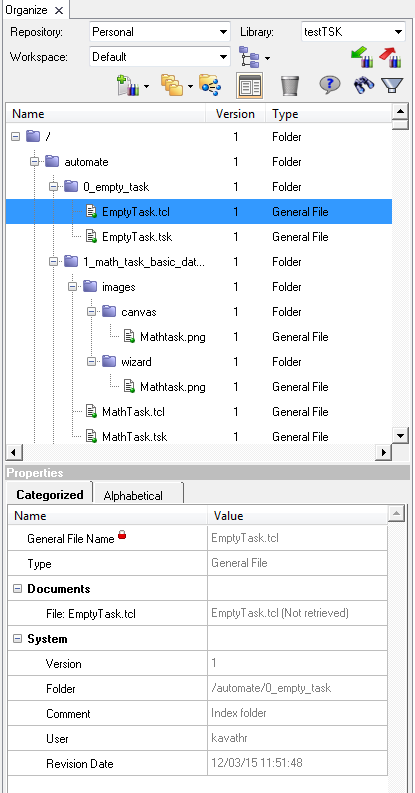


Fig 2.4 HyperWorks Collaboration Tools

**2.4 Workflow of existing modules in HyperWorks Automate**

**2.4.1 Existing Catalogues Sharing Procedure in HyperWorks Automate**

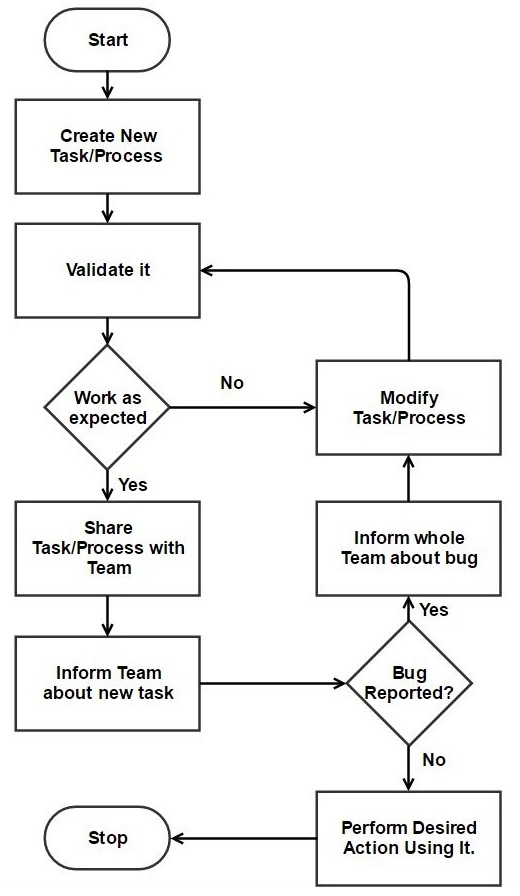
****

Fig. 2.5 Procedure to share Tasks, Processes or Catalogues

Figure 2.5 depicts procedure to share Tasks, Processes or Catalogues in HyperWorks Automate. Once the user creates any Tasks, Processes or Catalogues and want to share with team members then following step should be follows:

Step 1: Create New Tasks, Processes or Catalogues

Step 2: Validate it and if work as per expectation and go to step 3 else modify it.

Step 3: Share it via any methods like email/share directory etc. and inform team member.

Step 4: If any bug is found in task then inform all team member about bug, modify it and go

to step 2 and else perform desired action with it.

**2.4.2 Existing Debugging procedure in HyperWorks Automate**

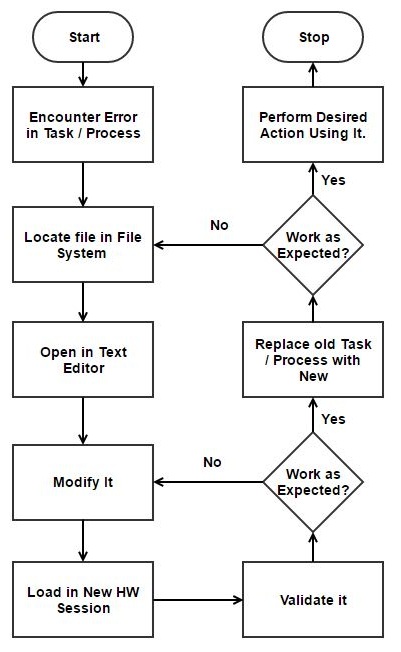


Fig. 2.6 Debugging in HyperWorks Automate

Figure 2.6 shows procedure to debug any Task or Process in HyperWorks Automate. While executing any task or process, if user encounters any error or bug in it, then user should follow the steps mentioned below:

**Step 1:** If user encounters any error/bug, then that task/process file location is located.

**Step 2:** Open it in any text editor and modify it.

**Step 3:** Load updated Task/Process in new session of HyperWorks Automate and validate it.

**Step 4:** If it works as expected than replace task/process with updated one else go to step 2.

**2.4.3 Build Verification Test (BVT) Infrastructure**

In software testing, a Build Verification Test (BVT), otherwise called Build Acceptance Test, is an arrangement of tests keep running on each new build of an item to check that the build is testable before the build is discharged under the control of the test group. In the software development, utilization of a BVT step is one sign of the consistent coordination development stage.

The build acknowledgment test is by and large a short arrangement of tests, which practices the standard usefulness of the application software. If any build successfully not execute then the build confirmation test is rejected.

Running a BVT before starting a full test run is vital on the grounds that it tells developers immediately if there is a major issue with the build, and they spare the test group squandered time and dissatisfaction by staying away from test of an unstable build. Figure 2.7 explains BVT execution flow.

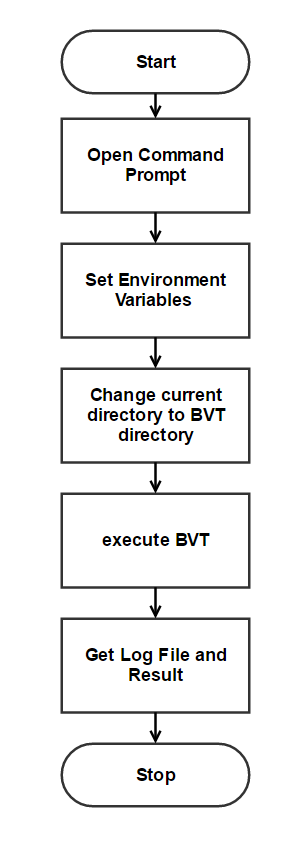


Fig. 2.7 BVT execution flow

Figure 2.7 shows existing procedure to execute BVT. For executing any BVT following steps should be followed:

**Step 1:** Open Command Prompt.

**Step 2:** Set Environment Variables (HW\_ROOTDIR, BVT\_ROOT, HW\_TARGET\_ARCH,

HW\_DEBUG).

**Step 3:** Change current directory to Desire BVT directory which user want to execute.

**Step 4:** Execute BVT using predefine execution command.

**Step 5:** Get Log File and Result file from BVT\_ROOT Directory.

**CHAPTER 3**

**METHODOLOGY**

This chapter first describes detailed description of methodology used for the project, then implementation details which includes detailed description about Database Persistence Module, Debugging Tool and BVT Infrastructure and at the end tools used for project work.

**3.1 Waterfall Software Development Methodology**

Waterfall methodology [9] is a first Software Development Life Cycle (SDLC) Model to be used comprehensively as a piece of Software Engineering to ensure accomplishment of the venture. In "The Waterfall" approach, the whole methodology of programming change is partitioned into independent phases. Waterfall model, normally, the aftereffect of one phase goes about as the contribution for the accompanying phase continuously. Figure 3.1 is representation of various stages in waterfall model.

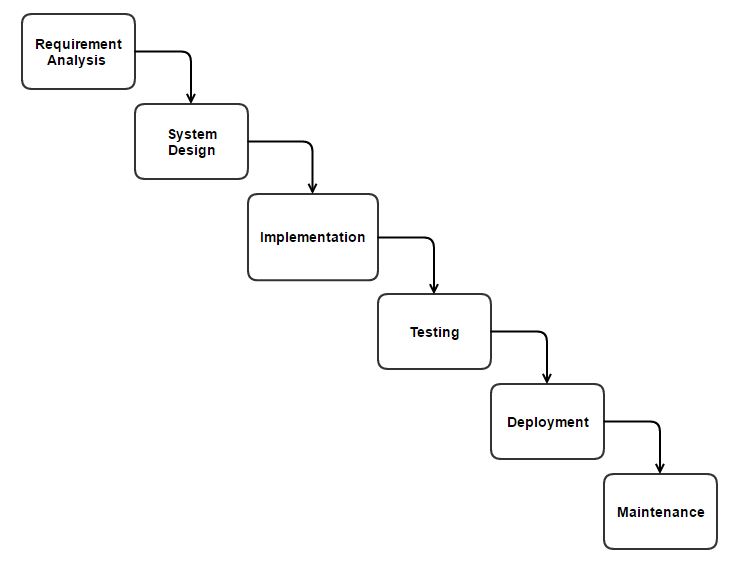


Fig. 3.1 Waterfall Model

* **Requirement Gathering and analysis:** In this stage, all possible requirements for system are collected from client and document it which is known as system requirement specification (SRS).
* **System Design:**  After getting SRS from first stage, System design is created. It will help to identify software and hardware required for project and also give idea about expected system structures.
* **Implementation:** Once system design is completed, initially small programs are created which are known as units. All units are individually tested for validating its functionality which is also known as “Unit Testing”.
* **Integration and Testing:** Every one of the units created in the implementation phase are joined into a system in the wake of testing of each unit. After integration entire system being tested to ensure proper functionally and compatibility with each other.
* **Deployment of system:** Once testing is finished, the end product is deployed in the client environment or product and the changes made are available the business sector.
* **Maintenance:** After deployment product in client environment, there are some issues which come up. To solve that issues patches are developed to deploy it in client environment. Also to enhance the product efficiency improvement are implemented. Backing is done to pass on these adjustments in the environment of customers.

All these phases mentioned above are connected to each other as shown in Figure 3.1. Next phase is started only after completing previous phase (That’s why it’s known as waterfall). And more than two phase cannot be executed simultaneously.

**3.2 Implementation Details**

**3.2.1 Implementation of Database Persistence Module**

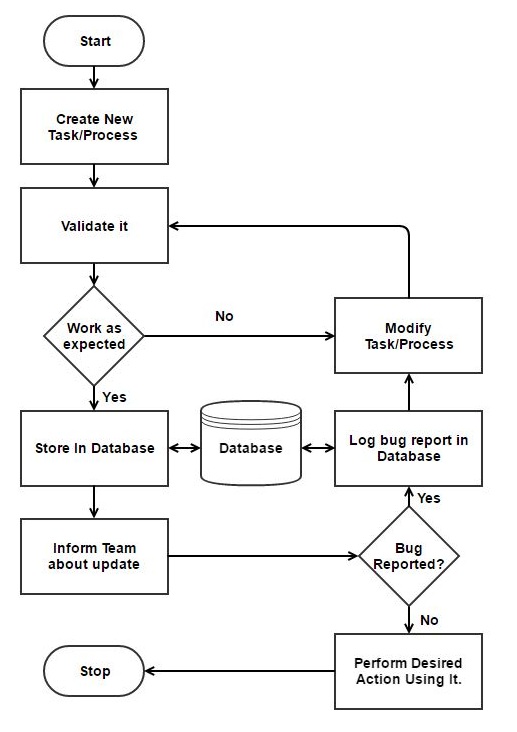
****

Fig. 3.2 Database Persistence Module for HyperWorks Automate

To achieve Database Persistence in HyperWorks Automate following step are followed:

* Development of Profile for HyperWorks Collaboration Tools
* Development of Parser
* Development of Tool to create Library
* Development of Tool to create Workspace
* Development of Module to Export Catalogue
* Development of Module to Import catalogue from Database

**3.2.1.1 Development of Profile for HyperWorks Collaboration Tools:**

Profile is a way to register non-core content types and any customization needed for the library to integrate Automate Documents and Files with Collaboration Tools.

It is a collection of content types, parsers and customizations using HyperWorks Desktop Collaboration Tools API's and Tcl/Tk [8] scripting which is needed to fulfil requirement as per user need. The following figure 3.3 displays content of profile.

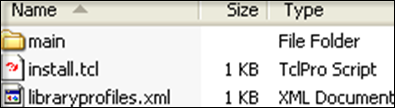


Fig 3.3 Content of Profile

As shown in figure 3.3:

* **Install.tcl** **File:**

This Tcl script is use to register created profile with HyperWorks Collaboration Tools so that all the needed non-core content types and any customization can be use from HyperWorks Collaboration Tools (Refer Figure 3.3).

* **Libraryprofiles.xml File:**

This “libraryprofiles.xml” file contains the path to content definitions, parsers, preferences, customization scripts and Tcl/Tk namespace for customization scripts (Refer Figure 3.3 & Figure 3.4).

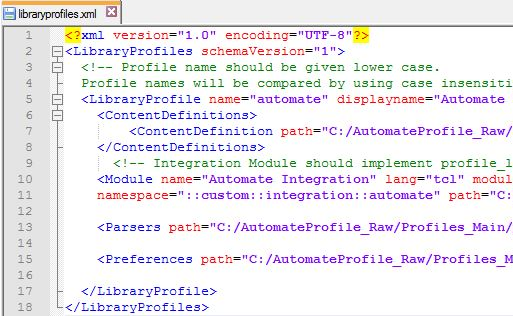


Fig 3.4 libraryprofiles.xml file content

* **Main Folder Content:**

Main folder contains the content definitions, parsers and customization scripts that are each organized in their respective folder. (Refer Figure 3.3 and Figure 3.5)

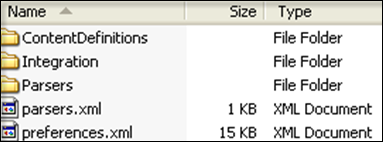


Fig 3.5 Main Folder Content

Figure 3.5 shows the content of Main Folder, which contains the following items:

* + **Content Definitions Folder:**

This folder contains all of the content definition in form of .xml files.

* + **Integration Folder:**

This folder contains Tcl/Tk scripts used for **Organize** browser customizations.

* + **Parsers Folder**:

This folder contains parser scripts used for parsing metadata from Automate Document.

* + **Preferences.xml File:**

This file give detail about preferences such as path to text editor, list of file types, and so on.

* + **Parsers.xml File:**

This file give list of parsers used for parsing metadata specific to the Automate Profile.

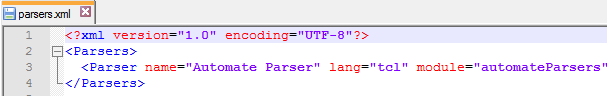


Fig. 3.6 Parsers.xml File Content

**3.2.1.2** **Development of Parser:**

This Parser extract the metadata from the Automate Document and Files, which parses the content to extract the metadata and store with file in data.

**This Parser does the following:**

* + Checks if the parsing is possible by verifying the file extension.
  + Parses the Automate file and extracts the metadata.
  + Adds the extracted metadata to the content.

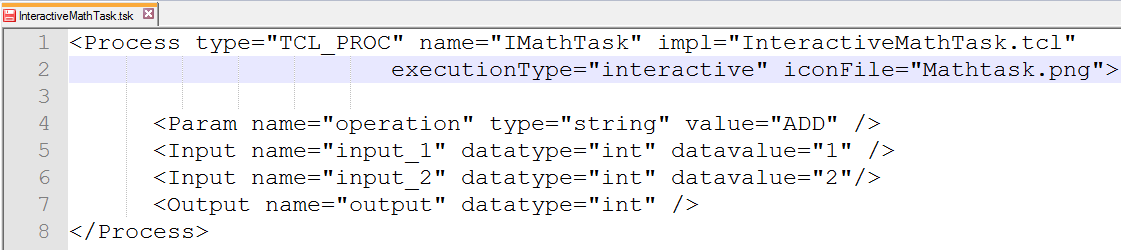


Fig 3.7 Example of Automate Document

Parser parses Automate Document (Let’s consider the Task file as shown in Fig. 3.7) and extracts metadata from Automate Document as Follows:

**Document Type** **:** TCL\_PROC

**Name of Process** **:** IMathTask

**Implementation File Location :** InteractiveMathTask.tcl

**Execution Type** **:** interactive

**Operation Perform** **:** ADD

**No. of Input** **:** 2

**Input Name** **:** “input\_1” “input\_2”

**No. of Output** **:** 1

**Output Name** **:** “output”

**Datatype** **:** int

After that, Parser stores this extracted metadata with content (Automate Document) in Database which is helpful for searching any Automate Document in Database.

**3.2.1.3 Development of Tool to create Library:**

This Tools create Library as per user’s need and user defined type which logically organizes or groups the contents stored within a repository. A repository is where data, information and associated files are located. When user wants to create library, user gives Library Name and Library Location to this tool, this tool then creates that library with given name and location. Then it registers this library with HyperWorks Collaboration Tools so the contents can be accessed from it. This tool is internally called by Export catalogue module.

**3.2.1.4 Development of Tool to create Workspace:**

A workspace is a local directory where the files are retrieved from the vault (Repository) and can be viewed and/or modified by user. The workspace maintains knowledge of what contents have been retrieved from the repository at any given time.

This tools create Workspace as provided by user and register that newly created Workspace with HyperWorks Collaboration Tools. Whenever any file retrieved to this Workspace, system store that detail so user get to know which files are already retrieved and among which files are updated. This tool internally call by Export catalogue module.

**3.2.1.5 Development of Module to Export Catalogue:**

Whenever any Automate user wants to export catalogue to database, this Export Catalogue module gives list of catalogues which are already registered with Automate. Among them, user is allowed to choose any or all catalogues to export to the database. After getting list of catalogues which user wants to export, this module will fetch each and every files registered in catalogues and parse with appropriate parsers and extract metadata from them and store those files in database and also add metadata with it, which will be helpful while searching for any Automate Process or Task from database.

**3.2.1.6 Development of Module to Import Catalogue from Database:**

Once user exports catalogue to database, it will to be shared between team members. Now there is a need to import that database in Automate. So that’s why this module is created. Import Catalogue Module allow user to fetch catalogue from database and display whole catalogue in HyperWorks Collaboration Tools by registering this database with it.

Now whenever any changes/modification is done in exported catalogue, it will be reflected to all the users of catalogues.

**3.2.2 Implementation of Debugging Module**

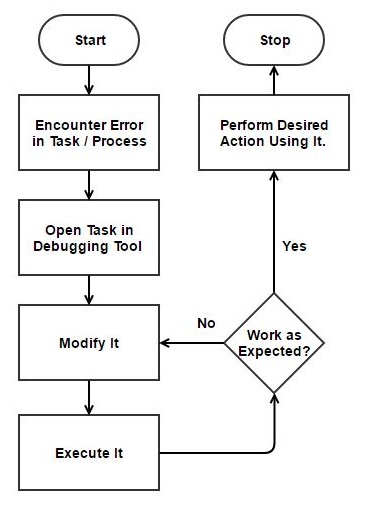


Fig. 3.8 Debugging Module for HyperWorks Automate

To implement debugging feature in HyperWorks Automate following tool has been developed:

* Inline Text Editor & Compiler to find Syntax error

**3.2.1.1 Development of Inline Text Editor & Compiler to find Syntax Error**

* For Inline Text Editor following methods are implemented as mention in table 3.1. Each method mention in table 3.1 perform basic task which required for text editor.
* For example, “SearchAndFindCommand” method search for desire words from file and Highlight it.
* “SearchAndReplace” method search for desire words from file and replace it with given words. Similarly all method will perform intended operation define in it.

Table 3.1 Methods defined for Inline Text Editor

|  |  |  |
| --- | --- | --- |
| AllListboxBindings | GoToLineGUI | SearchAndFindGUI |
| AllTextWidgetBindings | GoToLineNo | SearchAndFindNext |
| ArrayEditorValues | HandleDownArrowKey | SearchAndFindUp |
| AutoIndent | HandleUpArrowKey | SearchAndReplace |
| Close | Hide | SearchAndReplaceCommand |
| compileFileForErrorHandling | Init | SearchKeywords |
| CreateAutoCompleteUI | instantKeywordSuggestion | SearchKeyWordsAndHighLight |
| CreateListBox | LoadFile | SelectClosingBrace |
| Destroy | OnSelect | SelectedEndline |
| destroyAutoCompleteUI | Redo | SelectedStartline |
| EditFile | ReloadFile | SelectFromList |
| Execute | replaceCurrentWord | syntax\_highlight |
| FindText | ReplaceOne | tag\_word |
| GetActiveTextWidget | ResetPointerLocation | TestselectClosingBrace |
| getCurrentWord | ResetPointerLocationToEnd | TextCommenting |
| getCurrentWordIndex | SaveFile | ToggleHeight |
| getKeyWordList | ScriptFormat | Undo |
| getScreenCoordsForCurrentWord | SearchAndFindCommand | validate\_procedures |

**3.2.3 Implementation of BVT Infrastructure Module**

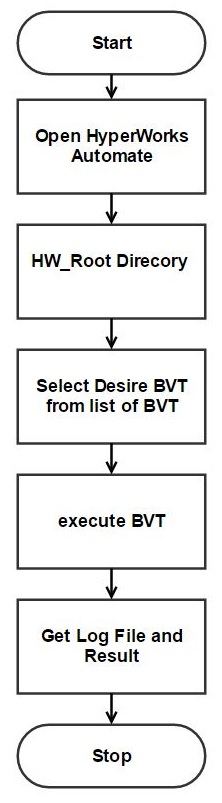


Fig. 3.9 BVT Infrastructure Module for HyperWorks Automate

* For BVT Infrastructure, various methods are implemented as mentioned in table 3.2. Each method mentioned in table 3.2 performs basic task required for BVT Infrastructure.
* For example, “createBVTInfraGUI” method creates GUI to get installation directory of HyperWorks and then searches for BVT directory and displays all BVT folders in Automate Browser in tree hierarchy.
* “RunBVTCall” method sets environment variables and then executes BVT. Once execution in completed, it will search for log files created by BVT and it changes BVT status as per log. Similarly all methods will perform intended operation defined in it.

Table 3.2 Methods defined for BVT Infrastructure

|  |  |  |
| --- | --- | --- |
| createBVTInfraGUI | loadBVTTab | RunBVTCall |
| DoesFileExistUnderDirectory | LoadBVTTree | searchforResultFile |
| ExecuteBVT | LoadTreeContent | serachForResultFileinBVTFolder |
| GetBVTNameFromCTRLfile | menuItems | showModelBrowserMenu |
| GUIFinalState | PlaceWindowInCentre | ValidateLoadBVTTree |
| GUIRunningState | publishHelp | waitforFinalResultFile |

**3.3 Tools Used**

**3.3.1 Perforce**

Perforce [6] is code repository in which code is to be checked after implementation and every alteration. Every user has to create his own local perforce branch on machine. After creating local branch Mapping is to be done between local space and development as well as production space

Following are some of the important operations which user can perform under perforce:

* Mark for add
* Mark for check out
* Submit
* Merge/Integrate
* Get full file and metadata history
* Difference between new code and previous version of code
* Change list which is generated every time when code is checked in or merged or integrated.

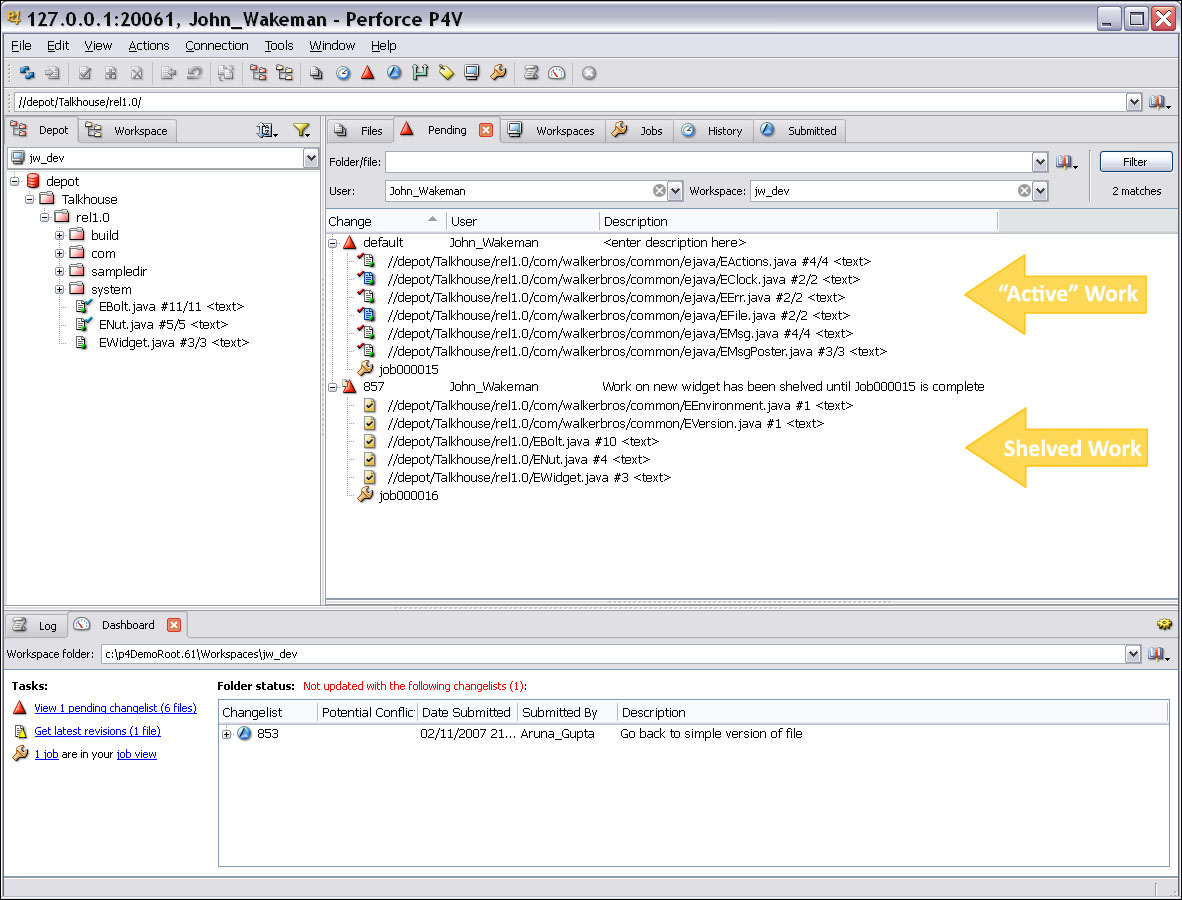


Fig 3.10: Snapshot of Perforce Client (Credit: www.perforce.com)

**CHAPTER 4**

**RESULT ANALYSIS**

**4.1 Introduction**

In this chapter, final outcome of all the modules mentioned in this report have been discussed.

**4.2 Result analysis**

**4.2.1 Database Persistence**

**4.2.1.1 Old Catalogue Sharing Approach**

* In existing approach, user needs to share new catalogues with team members through mail or directory sharing in network. As a result user has to inform each and every team member, that the catalogues have been added. If any bug is encountered then user needs to inform every team member that they need to restrict using the catalogues till changes are made. Once the changes are made users need to be re-informed and same process will be repeated.

**4.2.1.1 New Catalogue Sharing Approach**

* With the help of this module, without much effort team members can share their catalogues to each other. To access it, just click on either Import Catalogue or Export Catalogue button in HyperWorks Automate Toolbar as shown in Figure 4.1.

****

Fig 4.1 Snapshot of HyperWorks Automate Toolbar

* **Export catalogue** **Module:** At whatever point any Automate client need to Export catalogue to database, this Export Catalogue module give rundown of catalogues which are as of now enlist with Automate. Among them client permit to pick any or all catalogues to export to database. In the wake of getting rundown of catalogues which client need to export, this module will bring every single documents register in that catalogues and parse with fitting parsers and concentrate metadata from them and store those records in database furthermore include metadata with it, which will be useful while hunting down any Automate Process or Task from database. (Refer Figure 4.2)

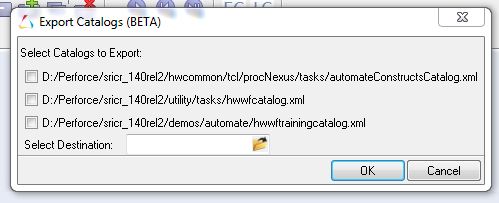


Fig 4.2 Snapshot of Export catalogue Module

* **Import catalogue** **Module:** When client export catalogue to database, it will to be offer between colleagues. Presently there is a need to import that database in Automate. So that is the reason this module is made. Import catalogue Module permit client to get catalogue from database and show entire catalogue in HyperWorks Collaboration Tools by enlisting this database with it. Presently at whatever point any progressions/change done in exported catalogue, naturally it will reflect to every one of the clients of catalogues. (Refer Figure 4.3)

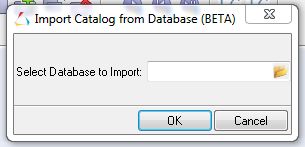


Fig 4.3 Snapshot of Import catalogue Module

**4.2.2 Debugging Module:**

**4.2.1.1 Old Debugging Approach**

* While executing any task or process if user encounters any error or bug in it, then user locates that task/process file location. Then user needs to open it in any text editor and modify it. After that user needs to load updated Task/Process in new session of HyperWorks Automate and validate it. If it works as expected then replace task/process with updated one. Otherwise relocate the file again and follow same procedure till it works as expected.

**4.2.1.1 New Debugging Approach**

* With the assistance of debugging feature in Automate will give capacity of finding and altering or bypassing bugs which makes Automate helpful and simple for individuals required with it.
* To access debugging module, user just right click on any task or process which user want to debug and select “Code View” option as shown in Figure 4.4.

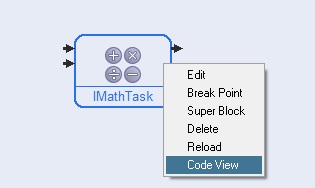
****

Fig. 4.4 Snapshot of IMathTask loaded in Automate canvas

* Once user click on “Code View”, code of task or process will open in Debugger Browser as shown in Figure 4.5.

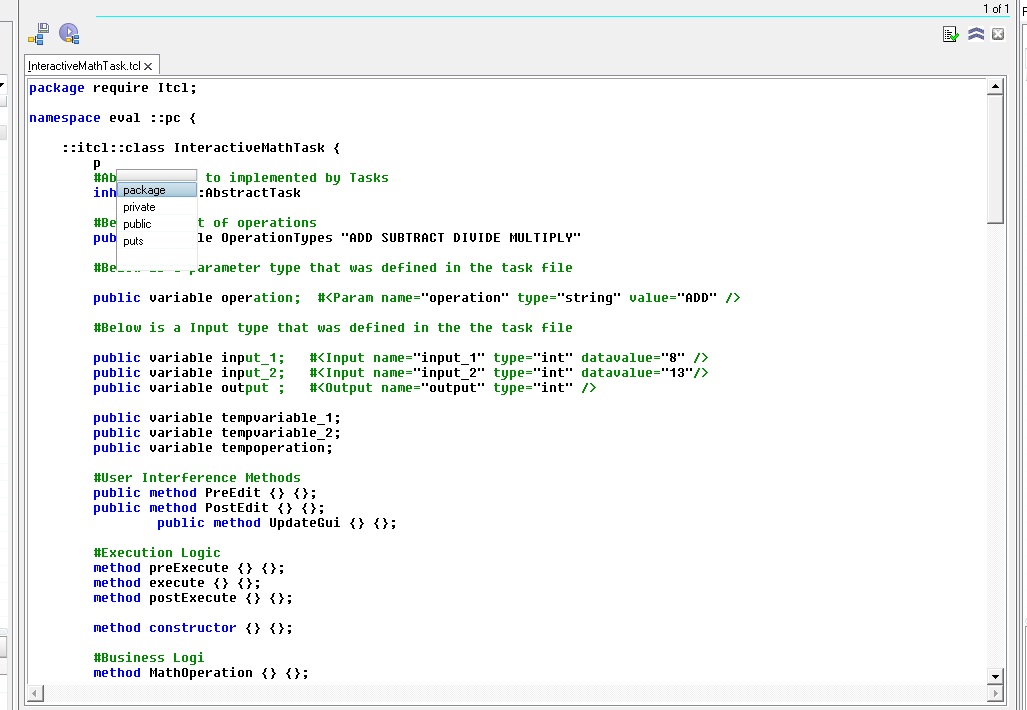
****

Fig. 4.5 Snapshot of IMathTask code open in Debugger Browser

* This Debugger Browser highlight all the syntax and also when user type any code it also give instant suggestion as shown in Figure 4.6.

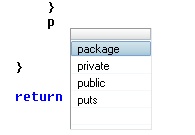


Fig. 4.6 Snapshot of Instant Suggestion given by Debugger Browser

* If user want to go to some particular line then by Pressing CTRL+G from keyboard, user can call Goto Line Module and give desire line number to go as shown in Figure 4.7.

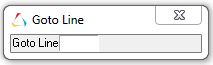


Fig. 4.7 Snapshot of Goto Line Module in Debugger Browser

* If user want to search for some string in code then by pressing CTRL+F from keyboard, user can call Find String Module and give desire string to search in either Up or Down Direction as shown in Figure 4.8.



Fig. 4.8 Snapshot of Find String Module in Debugger Browser

* If user want to search and replace some string in code then by pressing CTRL+H from keyboard, user can call Find & Replace String Module and can replace string with desire string in either Up or Down Direction as shown in Figure 4.9.



Fig. 4.9 Snapshot of Find & Replace String Module in Debugger Browser

**4.2.3 BVT Infrastructure:**

**4.2.3.1 Old BVT Execution Approach**

* In existing approach for BVT execution, users need to use command prompt and has to write complicated and length code as shown in figure 4.10.

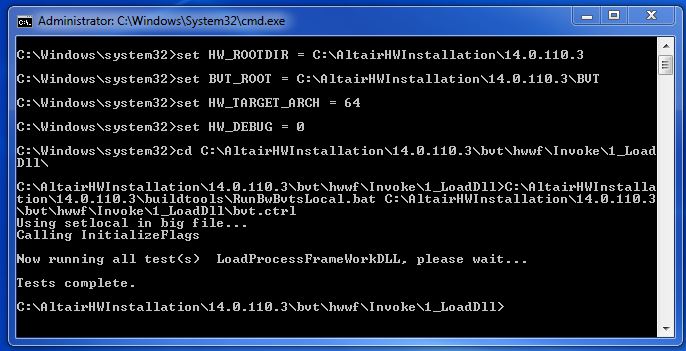


Fig. 4.10 Snapshot of old BVT Execution approach using command prompt

**4.2.3.2 New BVT Execution Approach**

* Introducing BVT Infrastructure in HyperWorks Automate has given new ability to execute any BVT directly from HyperWorks Automate without writing complicated and lengthy code to set environment variable in framework.
* To launch BVT Infrastructure, call “Launch Automate Tests Browser” from HyperWorks Automate. After successful call, a window will open as shown in Figure 4.11. Here user can input HyperWorks Installation directory for which user wants to execute BVT.

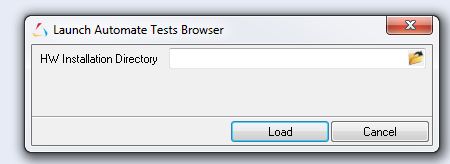
****

Fig. 4.11 Snapshot of Launch Automate Tests Browser

* Once user give HyperWorks installation directory and press Load button, BVT browser will open which contain all the BVT define in it as shown in Figure 4.12.

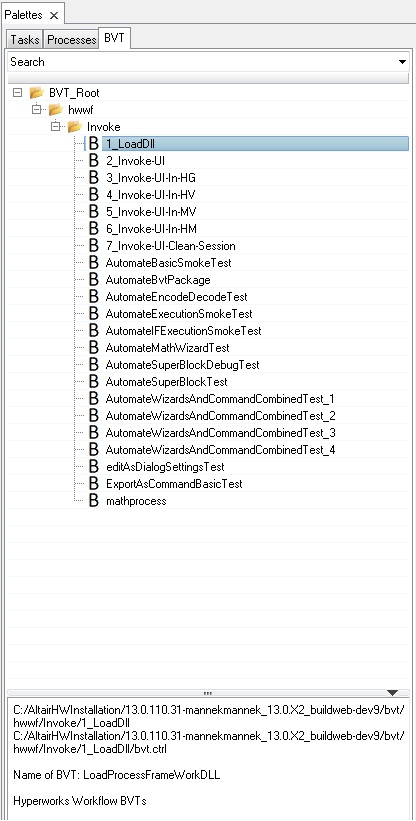
****

Fig 4.12 Snapshot of BVT Browser in HyperWorks Automate

* To execute any BVT just right click on it and then select “Run BVT” option as shown in Figure 4.13.

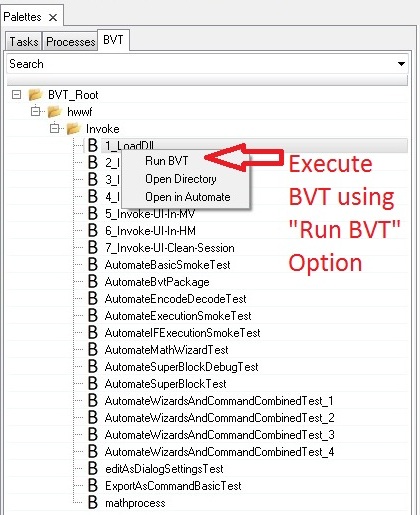


Fig 4.13 Snapshot of Run BVT Option In BVT Browser

* Once any BVT start executing, Execution Status of it change to Running State as shown in Figure 4.14, which guide about which BVT is currently executing in background of system.

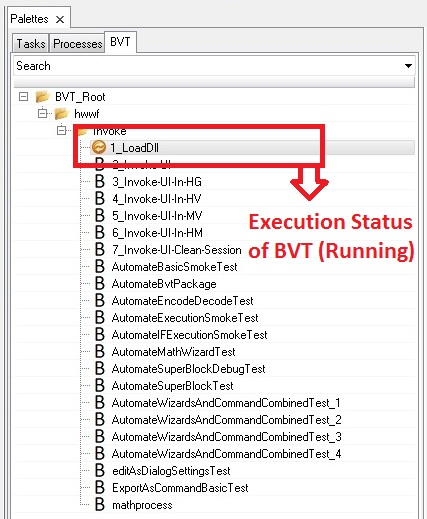


Fig 4.14 Snapshot of Execution Status of BVT (Running)

* If BVT successfully execute then Execution Status of that BVT convert to Success as shown in Figure 4.15.

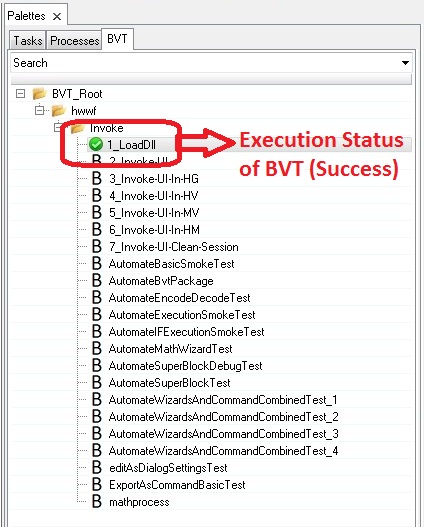


Fig 4.15 Snapshot of Execution Status of BVT (Success)

* If BVT not execute successfully then Execution Status of that BVT convert to Failure as shown in Figure 4.16.

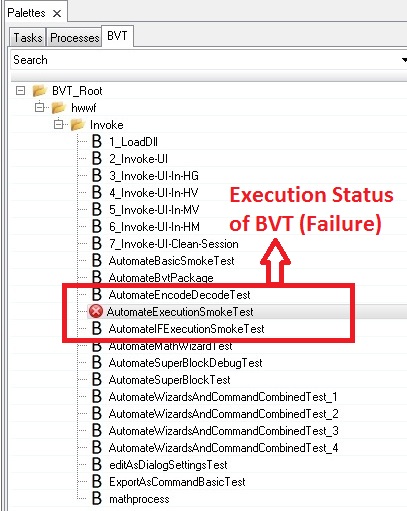


Fig 4.16 Snapshot of Execution Status of BVT (Failure)

* After successfully completion of BVT execution, final result will be open in default browser of system as shown in Figure 4.17 and Figure 4.18.

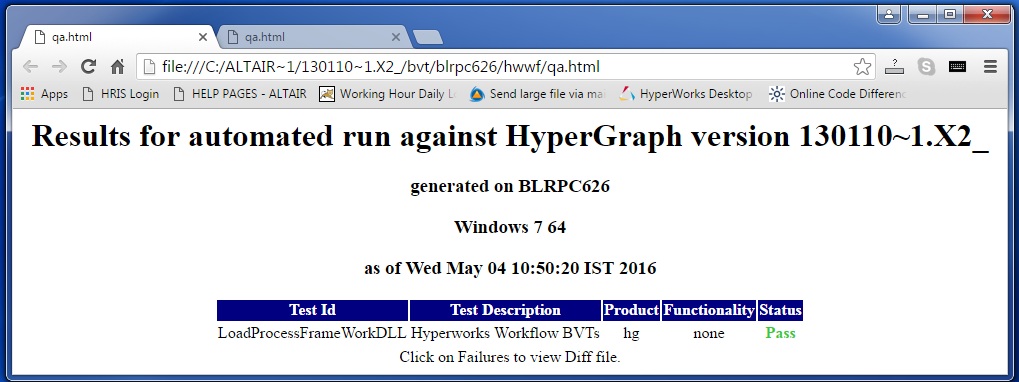


Fig 4.17 Snapshot of BVT Log Report (Success)

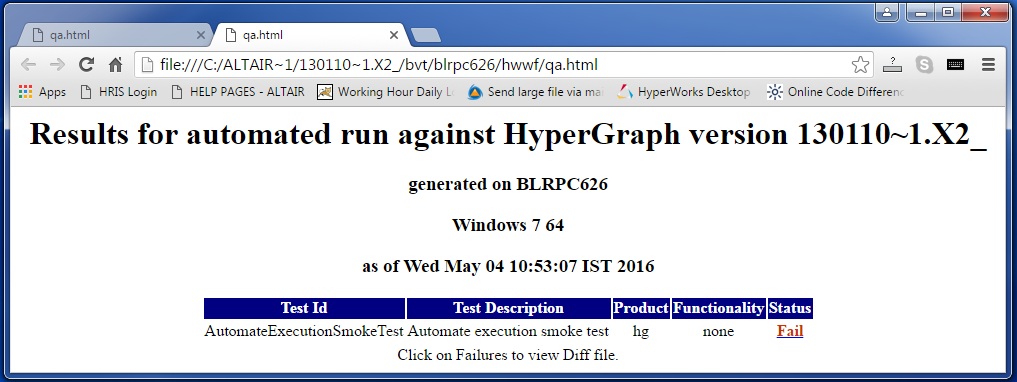


Fig 4.18 Snapshot of BVT Log Report (Failure)

**CHAPTER 5**

**CONCLUSION AND FUTURE SCOPE OF WORK**

**5.1 Conclusion**

To support constant enhancement in HyperWorks Automate these Database Persistence module, Inline Debugger Module and Test Frameworks for Tasks have been developed.

In order to facilitate Central Repository and Versioning functionality, Database Persistence module has been implemented, which enables the team members to save and share their catalogues in a central repository (which also supports version control) and also allow to search for tasks from database. With the help of this module user will get list of catalogues available in system and allow to store them in database at given location and also allow to access all the existence databases.

To ensure reliability and stability of Automate tasks, Test Framework for Tasks has been developed. Test Frameworks for Tasks allow mundane testing job to be automated. This framework display all the tests define in system and user can select any test to execute by clicking on execute option. Once execution is completed final result and log file will be available for validation.

For enhancing task developer’s productivity Inline Debugger have been developed. With the help of inline debugger, task developer can debug tasks during its development. This module allow to edit any task file by provide all basic facility of text editor like cut, copy, paste, undo, redo, find & replace and also highlight syntax error in code.

**5.2 Future Scope of Work**

Scope for future work would be to implement database persistence module capability in Test Framework for Tasks. The new capability could make it easier for people to share their Test Files with each other.

**REFERENCES**

1. Build verification test [Online].

Available: http://en.wikipedia.org/wiki/Build\_verification\_test [Accessed: May 10, 2016]

1. Debugger [Online].

Available: http://en.wikipedia.org/wiki/Debugger [Accessed: January 10, 2016]

1. Flow-based programming [Online].

Available: http://en.wikipedia.org/wiki/Flow-based\_programming [Accessed: November 11, 2015]

1. HyperWorks Automate [Online].

Available: http://altairhyperworks.com/automate [Accessed: August 10, 2015]

1. HyperWorks Collaboration Tools [Online].

Available: http://altairhyperworks.com/collaborationtools [Accessed: August 25, 2015]

1. Perforce [Online].

Available: http://en.wikipedia.org/wiki/Perforce [Accessed: September 14, 2015]

1. SQLite [Online].

Available: http://en.wikipedia.org/wiki/SQLite [Accessed: August 28, 2015]

1. Tcl/Tk [Online].

Available: https://www.tcl.tk [Accessed: May 01, 2015]

1. Waterfall Model [Online].

Available: http://www.tutorialspoint.com/sdlc/sdlc\_waterfall\_model.htm [Accessed: October 13, 2015]

PROJECT DETAILS

|  |  |  |  |
| --- | --- | --- | --- |
| *Student Details* | | | |
| **Student Name** | **KAVATHIYA HARSHAD RAMANIKLAL** | | |
| Register Number | 140913004 | Section / Roll No | CSE /140913004 |
| Email Address | harshad3011@gmail.com | Phone No (M) | +91-8866379912 |
|  | | | |
| *Project Details* | | | |
| **Project Title** | **DEVELOPING PLUGINS FOR ALTAIR HYPERWORKS AUTOMATE (A PROCESS AUTOMATION FRAMEWORK)** | | |
| Project Duration | 10 months | Date of reporting | 3rd August, 2015 |
|  |  | | |
| *Organization Details* | | | |
| **Organization Name** | **Altair Engineering India Pvt. Ltd.** | | |
| Full postal address with pin code | Mercury 2B Block, 5th Floor, Prestige Tech Park,  Sarjapur Marathahalli, Outer Ring Road, Bangalore – 560 087  Tel: +91-80-6629 4500 Fax: +91-80-6629 4700 | | |
| Website address | www.altair-india.in | | |
|  |  | | |
| *Supervisor Details* | | | |
| **Supervisor Name** | **Mr. Deepak Kumar** | | |
| Designation | Senior Manager | | |
| Full contact address with pin code | Altair Engineering India Pvt. Ltd.  Mercury 2B Block, 5th Floor, Prestige Tech Park, Sarjapur Marathahalli, Outer Ring Road, Bangalore – 560 087 | | |
| Email address | deepak.kumar@altair.com | Phone No (M) | +91-9448357155 |
|  |  | | |
| *Internal Guide Details* | | | |
| **Faculty Name** | **Dr. Renuka A** | | |
| Full contact address with pin code | Dept. of Computer Science & Engg., Manipal Institute of Technology, Manipal – 576 104 (Karnataka State), INDIA | | |
| Email address | renuka.prabhu@manipal.edu | | |