|  |
| --- |
| R2/R3 System Release Build Process |
|  |

**Abstract**

This document contains R2/R3 Release Environment description

Table of CONTENTS

[1. Revision history 4](#_Toc411254427)

[2. Introduction 5](#_Toc411254428)

[2.1 Purpose 5](#_Toc411254429)

[2.2 Abbreviations 5](#_Toc411254430)

[3. IMPORTANT PATHS AND LINKS 6](#_Toc411254431)

[4. SERVER INFORMATION 7](#_Toc411254432)

[5. BUILD FRAMEWORK 8](#_Toc411254433)

[5.1 Tools/Programming Languages used in CMIS 8](#_Toc411254434)

[5.2 CMIS 9](#_Toc411254435)

[6. WORK FLOWS 11](#_Toc411254436)

[6.1 Linux Based Build Procedure(FrontEnd) 11](#_Toc411254437)

[6.2 Linux Based Build Procedure(BackEnd) 15](#_Toc411254438)

L**ist of Tables**

**No table of figures entries found.**Insert legal content (ownership, trademarks, etc) that shall be in this type of document.

Response on required legal appreciated. Use company standard if any.

This header will be outside the table of contents, TOC.

For headings outside the TOC use format template”Title1”, ”Title2” or ”Title3”. Please refer to the example ”Legal heading 1” above. Please erase this hidden text when done reading.

# Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Doc Version | Date | By | Description |
| 0.1 | 06-Feb-2015 | Anubhav | First VNL Products Build Framwork Document |
|  |  |  |  |
|  |  |  |  |

# Introduction

This Document explains all the Build process of VNL project and its environment with details along with GUI screen.

## Purpose

This document provides completes description of the R2/R3 Release making process and environment applicable.

## Abbreviations

|  |  |
| --- | --- |
| SVN | Subversion |
| CMIS | Configuration Management Information System |
| GUI | Graphical User Interface |
| RC | Release Candidate |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# IMPORTANT PATHS AND LINKS

* <http://10.100.211.65/BSS_Releases/CM_Docs/>[Build Environment Process document.docx](http://10.100.211.65/BSS_Releases/CM_Docs/Build Environment Process document.docx)
* <http://10.100.211.61/test/cm-test/CM-TEST/trunk/Vivek/ROOT/Windows_compliation_Bakup/Windows_compilation/>
* http://10.100.211.65/BSS\_Releases/CM\_Docs/[CMIS Overview.pptx](http://10.100.211.65/BSS_Releases/CM_Docs/CMIS Overview.pptx)
* http://10.100.211.65/BSS\_Releases/CM\_Docs/ [Subversion\_BasicTutorials.doc](http://10.100.211.65/BSS_Releases/CM_Docs/Subversion_BasicTutorials.doc)
* http://10.100.211.65/BSS\_Releases/CM\_Docs/ [VersionControlSystem.ppt](http://10.100.211.65/BSS_Releases/CM_Docs/VersionControlSystem.ppt)

# SERVER INFORMATION

|  |  |  |
| --- | --- | --- |
| MACHINE NAME | MACHINE IP | USAGE |
| BHARAT | 10.100.211.63 | It Holds CMIS(Build Framework GUI) code.All type of java and .jsp files. |
| INDIA | 10.100.211.65 | It Keeps all code SVN Repositry. |
| HINDUSTAN | 10.100.211.61 | Main Compliation Server. |
| BUILD-SERVER-FOR-DATA-PRODUCTS | 10.100.211.68 | All Data Products like RU,AP etc. Compiling. |
| R3-BUILDSERVER | 10.100.211.74 | All R3 Build like R3-Platform,R3-Stack etc, |
| CMADMIN | 10.100.208.110 | All Windows compliation like R2-BTS,EDS,TDN,CCU,FRFC,RARU etc. |
|  | 10.100.211.85 | R3-BTS compilation. |

# BUILD FRAMEWORK

Build Framework contain all the build procedure of different kind of build which include different environment like Linux, Ubuntu and Windows. It includes GUI and script part which make complete package of Build Framework. Here explain all the different component of Build Framework.

## Tools/Programming Languages used in CMIS

* **apache-tomcat-6.0.29.** used to deploy/run CMIS’s source code. Or we can use BEA weblogic , IBM WebSphere etc.
* **Java-mail.** used to send mail to developers based upon project name including us and requestor.
* **AJAX (Asynchronous JavaScript And XML).** AJAX = Asynchronous JavaScript and XML.AJAX is a technique for creating fast and dynamic web pages. AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

Classic web pages, (which do not use AJAX) must reload the entire page if the content should change.

Examples of applications using AJAX: Google Maps, Gmail, Youtube, and Facebook tabs.

* **JSP (java server pages).** JavaServer Pages (JSP) is a Java technology that helps software developers serve dynamically generated web pages based on HTML, XML, or other document types.

Architecturally, JSP may be viewed as a high-level abstraction of Java servlets. JSP are loaded in the server and are operated from a structured special installed Java server packet called a Java EE Web Application, often packaged as a .war or .ear file archive.

JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, with the resulting page being compiled and executed on the server to deliver an HTML or XML document. The compiled pages and any dependent Java libraries use Java bytecode rather than a native software format, and must therefore be executed within a Java virtual machine (JVM) that integrates with the host operating system to provide an abstract platform-neutral environment.

* **Java script.** JavaScript is a prototype-based scripting language that is dynamic, weakly typed and has first-class functions. It is a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles.

JavaScript was formalized in the ECMAScript language standard and is primarily used in the form of client-side JavaScript, implemented as part of a Web browser in order to provide enhanced user interfaces and dynamic websites. This enables programmatic access to computational objects within a host environment.

JavaScript's use in applications outside Web pages — for example in PDF documents, site-specific browsers, and desktop widgets — is also significant.

* **Servlet.** A servlet is a Java programming language class used to extend the capabilities of servers that host applications accessed via a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by Web servers. Thus, it can be thought of as a Java Applet that runs on a server instead of a browser.

A Servlet is a Java class in Java EE that conforms to the Java Servlet API, a protocol by which a Java class may respond to requests. They are not tied to a specific client-server protocol, but are most often used with the HTTP protocol. Therefore, the word "Servlet" is often used in the meaning of "HTTP Servlet“. Thus, a software developer may use a servlet to add dynamic content to a Web server using the Java platform. The generated content is commonly HTML, but may be other data such as XML. Servlets are the Java counterpart to non-Java dynamic Web content technologies such as CGI and ASP.NET. Servlets can maintain state in session variables across many server transactions by using HTTP cookies, or URL rewriting

## CMIS

CMIS which include the entire GUI part which display for user. The first screen display here and its available on [cmis.vnl.in](http://10.100.211.63/) (VNL networks only).

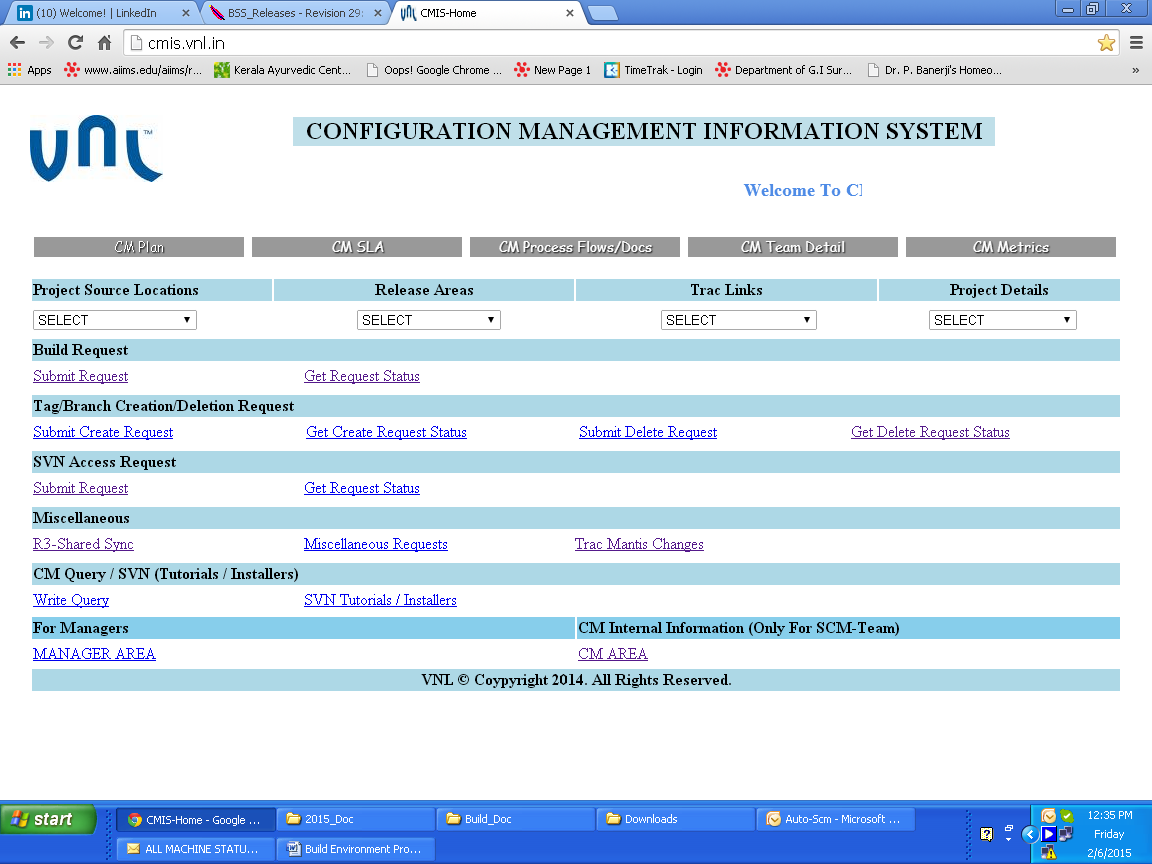


Figure 1

The following links will help u for explore the feature of CMIS.

* **Project Source Locations.** U can explore the code Repositry here.
* **Relese Area.** U can explroe all the RC release area.
* **Track Link.**  Track the particular Repostiory with all the history.
* **Project Details.** This will display all information about project like Repo location, relese location ,log location etc.
* **Build Requst.** This option use fire the DEV-Build or RC Build by any of the user( which have to right for fire the release).
* **Get Request.** User Can get the status of fired build with the help of id.
* **Submit Create Request.** For creating Tag/Branches request.
* **Get Create Request Status.** Verfiy the raised request status.
* **Submit Request(SVN Access).** This option will use for taking access(read/write) on SVN repositry.
* **Get Request Status(SVN Access).** This option will track your submited request with the help of ID.

# WORK FLOWS

This Section will explain about the complete work flow from end to end of build procedure including CMIS and complete build script framework. Here will explain about each type of build procedure like Linux, Ubuntu and Windows.

## Linux Based Build Procedure(FrontEnd)

Here will take example of PS or BSC (that have maximum dependencies with different build).

**Step 1** the option select Submit request from [cmis.vnl.in](http://10.100.211.63/) using your linux username/password.



Figure 2

**Step 2**  select PS from list.

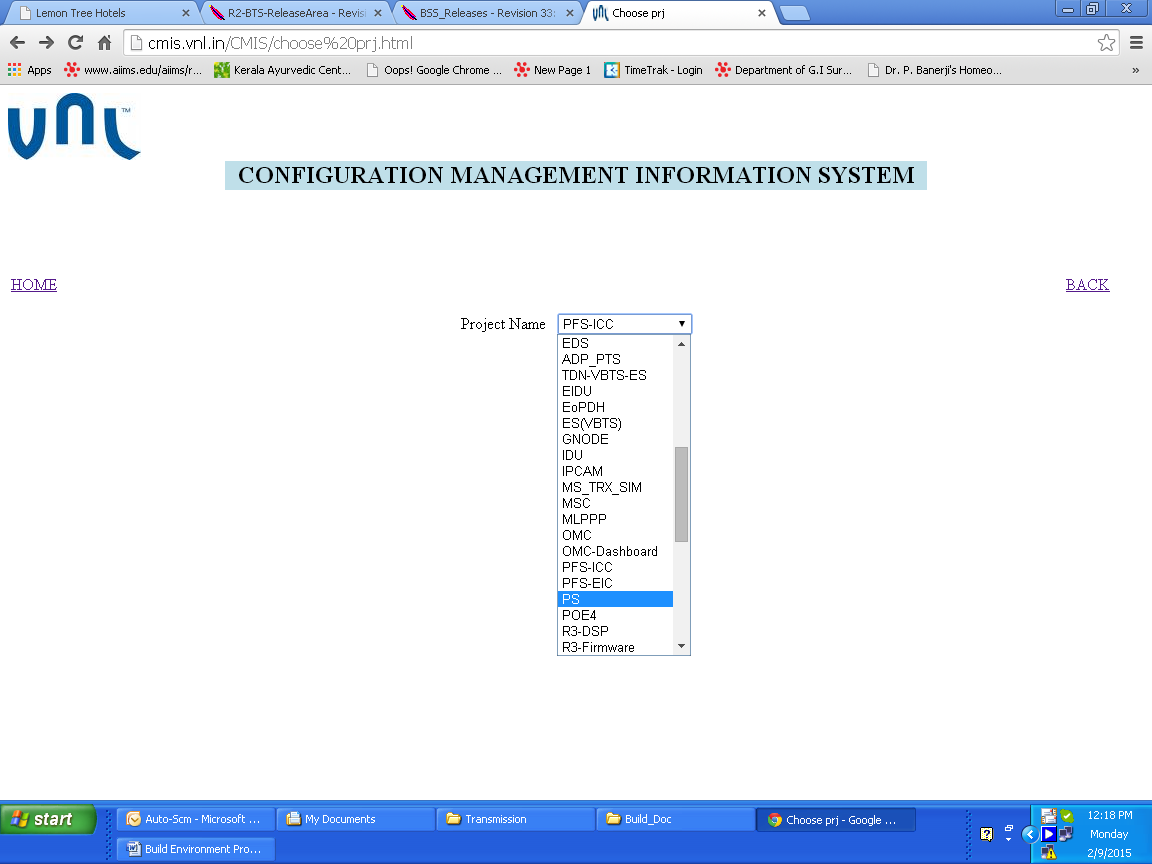


Figure 3

Step 3 Select DEV-Build or RC from list (RC and DEV-Build is only diffrence when release goes to user for Dev testing called DEV-Build or Release goes to Validation Called RC).

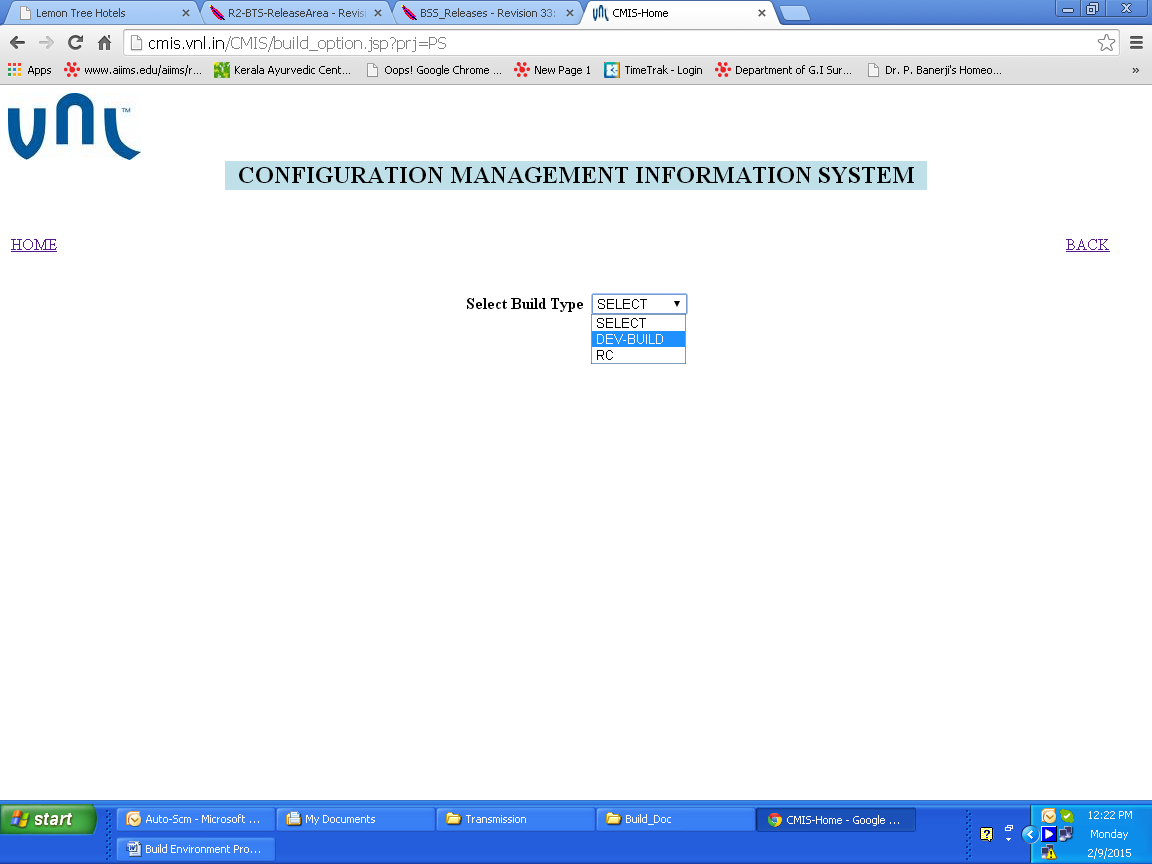


Figure 4

**Step 4** User can select the branch,revision no,release version,BPM version,PFS-ICC version,PFS-EIC version,DSP version,DSP-BSC-AbisoIP on this form.

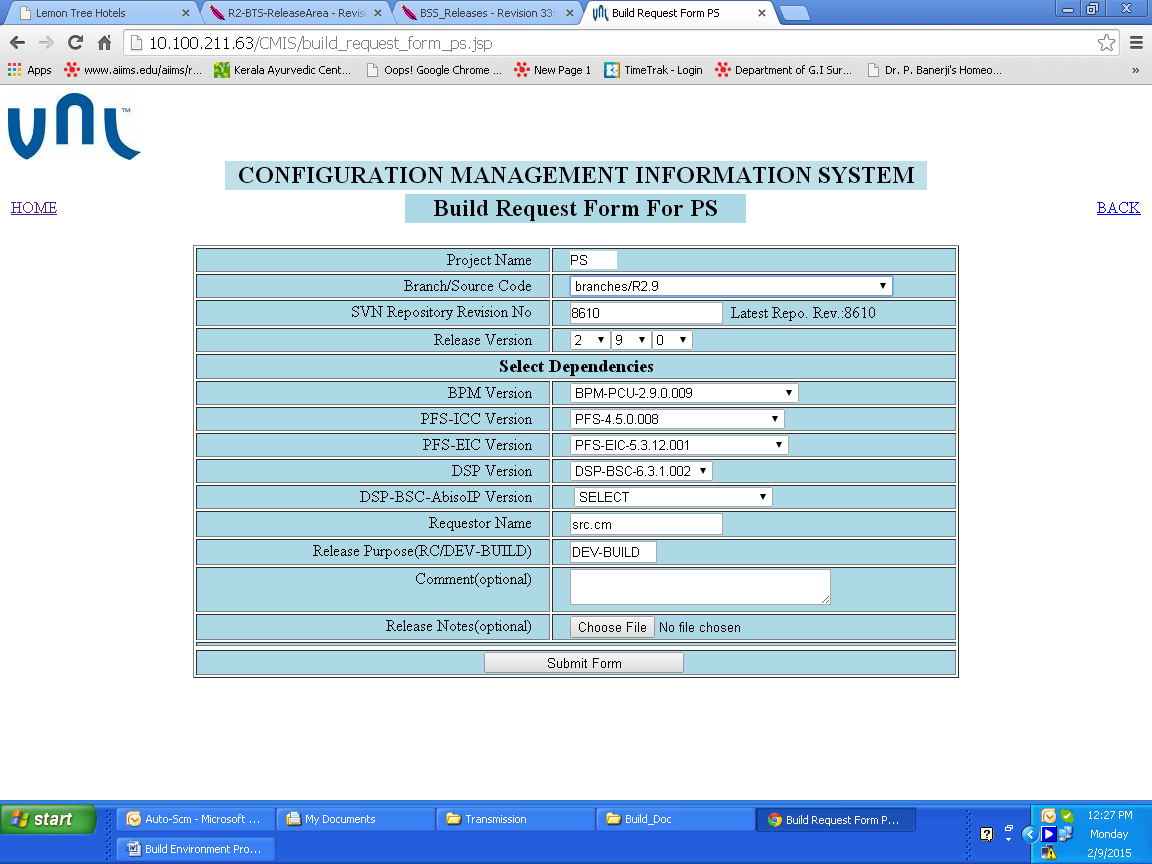
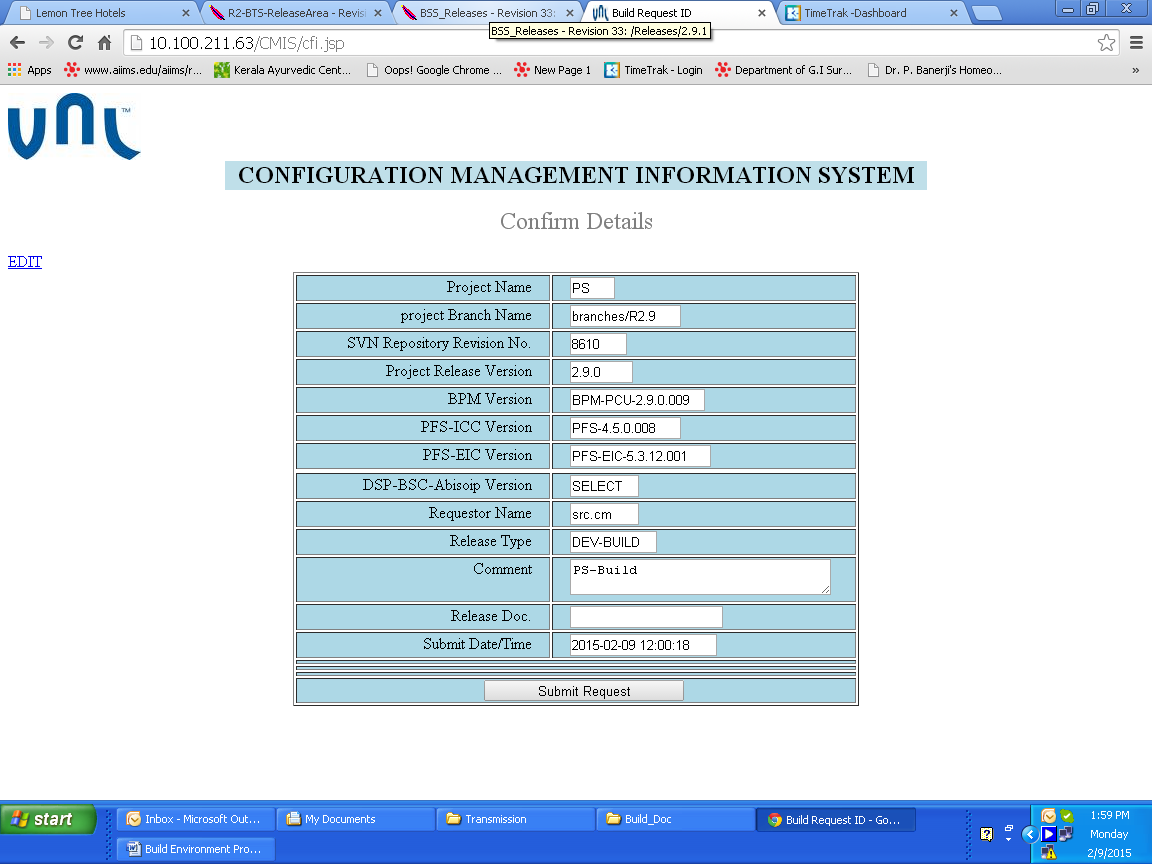


Figure 5

**Step 5** This page will display all the final parameter which selectd by the user and with these parameter User going to make DEV-Build/RC Build. When u Submit Request all the concern person will get starting mail.



## Linux Based Build Procedure(BackEnd)

This Section will explain about the Backend procedure when any build fired.

The following points will automatically perfome when any build fired.

1. All the build information like Project name, branch, revision no ,dependencies will reach on **build.ksh(/home/cmbuild/Build\_FRMWK/ROOT/)** file on **10.100.211.61( main compilation server)** server.
2. **build.ksh** script file contain all the project related stuff like mandatory parameters etc. all the parameters insert into different variable which will use for further processing.
3. Every project have **.cfg** file and **.ksh** file like project VNL-NIB have VNL-NIB**.cfg** (/home/cmbuild/Build\_FRMWK/ROOT/config/project-specific/ VNL-NIB) and **VNL-NIB\_COMMON\_BUILD.ksh** (/home/cmbuild/Build\_FRMWK/ROOT/script/project-specific/VNL-NIB).
4. **.cfg** file will contain all the project related configuration like, Repo name, Repo URL, user mail ids, OS type, start date, end date etc.
5. **.ksh** file contain all the build related procedure like, code checkout, set environment variable, dependency code check out, compiling code and package into **.tar** file or **.zip** file.
6. **.ksh** file can transfer build on some other machine using **ssh** command if required. It can transfer file in Ubuntu, windows based and some other Linux server.

## Windows Based Build Procedure(BackEnd)

This Section will explain about the Backend procedure when any build fired.

The following points will automatically perfome when any build fired.

1. All the build information like Project name, branch, revision no, dependencies will reach on **build.ksh(/home/cmbuild/Build\_FRMWK/ROOT/)** file on **10.100.211.61( main compilation server)** server.
2. **build.ksh** script file contain all the project related stuff like mandatory parameters etc. all the parameters insert into different variable which will use for further processing.
3. Every project have **.cfg** file and **.ksh** file like project EDS have **EDS.cfg** (/home/cmbuild/Build\_FRMWK/ROOT/config/project-specific/EDS) and **EDS\_COMMON\_BUILD.ksh** (/home/cmbuild/Build\_FRMWK/ROOT/script/project-specific/EDS).
4. **.cfg** file will contain all the project related configuration like, Repo name, Repo URL, user mail ids, OS type, start date, end date etc.
5. **.ksh** transfer build from 10.100.211.61 to 10.100.208.110(**window based compilation server**). All the parameters passed from 10.100.211.61 to 10.100.208.110 and window server file contain all the build related procedure like, code checkout, set environment variable, dependency code check out, compiling code and package into **.tar** file or **.zip** file.
6. **.ksh** file can transfer build on some other machine using **ssh** command if required. It can transfer file in Ubuntu, windows based and some other Linux server.

## Ubuntu Based Build Procedure(BackEnd)

This Section will explain about the Backend procedure when any build fired.

The following points will automatically perfome when any build fired.

1. All the build information like Project name, branch, revision no, dependencies will reach on **build.ksh(/home/cmbuild/Build\_FRMWK/ROOT/)** file on **10.100.211.61( main compilation server)** server.
2. **build.ksh** script file contain all the project related stuff like mandatory parameters etc. all the parameters insert into different variable which will use for further processing.
3. Every project have **.cfg** file and **.ksh** file like project RU have **RU.cfg** (/home/cmbuild/Build\_FRMWK/ROOT/config/project-specific/RU) and **RU\_COMMON\_BUILD.ksh** (/home/cmbuild/Build\_FRMWK/ROOT/script/project-specific/RU).
4. **.cfg** file will contain all the project related configuration like, Repo name, Repo URL, user mail ids, OS type, start date, end date etc.
5. **.ksh** transfer build from 10.100.211.61 to 10.100.211.68(**ubntu based compilation server**). All the parameters passed from 10.100.211.61 to 10.100.211.68 and ubntu server file contain all the build related procedure like, code checkout, set environment variable, dependency code check out, compiling code and package into **.tar** file or **.zip** file.
6. **.ksh** file can transfer build on some other machine using **ssh** command if required. It can transfer file in Ubuntu, windows based and some other Linux server.