



# Client Side Customization For Importing Legacy Data And Workflow Inprocess Objects In Plm Teamcenter-12.0

<sup>1</sup> Mr. Amit Bhelonde, <sup>2</sup> Mr. Sayyad Shafik,

<sup>1</sup>M.Tech in Mechanical Engineering, <sup>2</sup>Assistant Professor,  
<sup>1,2</sup>Department of Mechanical Engineering,  
<sup>1,2</sup>MPGI School of Engineering, Nanded, Maharashtra, India

**Abstract:** In present days of increasing software development environment sometimes it turns into challenging to mold the Product Lifecycle Management software as per requirement of clients. Most of the time user requires some simplified applications for its business solution and simplified process for their organization. To solve this problem client side customization is necessary means customize the client User Interface by using the Rich Architecture Customization and also used different Application Program Interface and Java language. Product Lifecycle Management is mainly concern about the data and through the Product Lifecycle Management we can integrate the data, people, processes and business systems. My aim is to import Legacy Data i.e. raw data present in form of physical files with different extensions to be imported in Teamcenter with same name of file to be given for Item Name, Revision Name and Dataset and I have also exported Workflow Inprocess Objects with Item ID, Item Name and Process Stage List i.e. Status from Teamcenter into MS Excel format. For this work I have used Teamcenter 12.0 along with Eclipse Photon tool, for client side customization. Data Exportation and Importation is an important activity in real-time implementation of Product Lifecycle Management which can be achieved through default functionality of PLM Teamcenter in xml format only. So, to export and import data according to client requirement in any format we have to use customization in PLM Teamcenter using API's provided by Siemens for client customization. By using the Eclipse plug-in project, I have made Swing Custom User Interface from that I can select file path in my computer where report is to be generated and by clicking export button data is exported in MS Excel format from Teamcenter.

**Keywords:** Product Lifecycle Management, Rich Application Client Customization, Teamcenter 12, Eclipse Based Plug in Project, User Interface.

## I. INTRODUCTION

Product Lifecycle Management frameworks offer assistance organizations in adapting with the expanding complexity and designing challenges of creating unused items for the worldwide competitive markets. Product Lifecycle Management ought to be recognized from product life-cycle Management (marketing). PLM depicts the designing perspective of a product, from overseeing depictions and properties of a product through its improvement and valuable life; while, PLM alludes to the commercial administration of life of a product within the commerce showcase with regard to costs and deals measures. Product Lifecycle Management is the method of overseeing the whole lifecycle of a product from initiation, through designing plan and make, to benefit and transfer of made products and also PLM manages data from raw material to finished products [1]. Product Lifecycle Management integrates data, people, processes and business systems and provides a product

information backbone for companies and its enterprise. Product Lifecycle Management tool has wider scope and there are various types of Product Lifecycle Management tools that integrate the data according to company needs. Companies have decided which PLM software is better but now days Siemens PLM has covered 70% of global competitive markets [2]. Today's point of view Teamcenter has covered 70% of the global market. There are lot of Product Lifecycle Management tools available in the market such as Siemens Teamcenter, PTC Windchill, Dassault ENOVIA, SAP PLM, Oracle Agile, Aras PLM, Arena PLM, Fuse PLM, Bamboo Rose, Autodesk Vault, Autodesk Fusion Lifecycle but Siemens Teamcenter is very popular among all of this due its flexibility and user friendly interface and also lot of Out of the Box (OTB) are available in the Teamcenter.

## II. PLM CUSTOMIZATION

### 2.1 Need of PLM

Product Lifecycle Management can be utilized to build yield with consistent assets, to expand incomes or to diminish the assets used to deliver a steady yield. This all assists with improving the reality. Product Lifecycle Management encourages association to accomplish this through: Productivity upgrades, Improving advancement for new business objects (products), Decreased expenses, Increment efficiency, improved nature of business objects (products).

Integration of data is the need of global market also integrated data consumes the memory so market needs integrate data with least consumable memory. From that concern has discovered the concept of PLM and its tools.

### 2.2 Objective

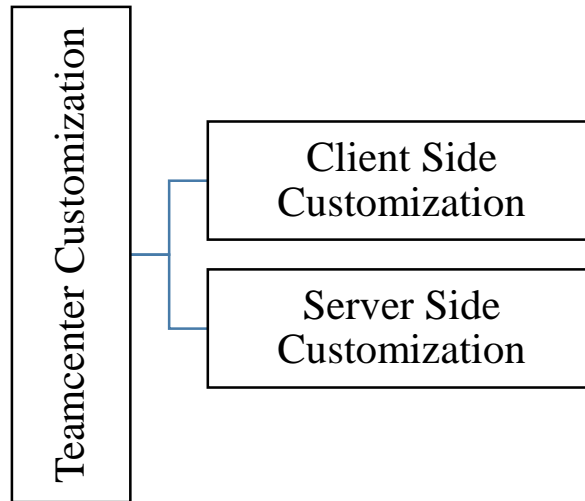
This research work has developed the solution for exporting data in MS Excel format from Teamcenter where we can implement Product Lifecycle Management in any organization.

This work has main objective of Exporting Data from Teamcenter 12 by using Rich Client Customization means by using Client Side Customization.

### 2.3 Types of Customization

#### 2.3.1 Client Side Customization

The client side customization can be performed by using programming language means Java and various wizards available in Teamcenter. This type of customization can be used for including new Application in Teamcenter, such as Menu bar, Toolbar, User Interface form on client side. By using this type of customization we can easily solve the customer requirement.



**Fig -1:** Teamcenter Customization types

### 2.3.2 Server Side Customization

The server is being customized using Teamcenter API (Application program interface) called as Integrated Tool Kit (ITK), C++. Integrated Tool Kit (ITK) is a set of software tools which you can use to integrate third party or user-developed applications with Teamcenter and we can use this customization for server side issues.

## 2.4 Customization Tools

Customization in Teamcenter can be performed in different ways and we have seen below [3].

### 2.4.1 Non-programming Customization

Non-programming customization can be performed by using the wizards available in Teamcenter and making the entries in the registry files.

### 2.4.2 Programming Customization

Programming customization can be performed by using programming languages i.e. C, C++, Java. This can be classified into server side and client side customization. Client side is customized using Java language and server side is by using ITK.

### 2.4.3 ITK (Integration Tool Kit)

The Integration Toolkit (ITK) is a set of server-side software tools that you can use to integrate third-party or user-developed applications with Teamcenter. The ITK is a set of C and C++ functions used directly by Teamcenter and UNIGRAPHICS. The Integration Toolkit is a set of software tools that act as a programmatic interface to Teamcenter. It is the means by which both internal and external applications integrate with the Teamcenter. Internal applications are those supplied such as Unigraphics. External applications (third party) are those that you decide to integrate into Teamcenter.

## 2.4.4 Portal Customization

Portal customization is performed by using Java programming. This can be used to add the new application or customize the menu bar and toolbar within Teamcenter engineering application.

## III. RESEARCH METHODOLOGY

For implementation of Product Lifecycle Management in an organization, the main concern is data so, we have to manage the data. Data export in other format rather than xml is the key phenomenon so I have exported data using Rich Application Client Customization. I have made custom dialog in Eclipse plug in project and names of the projects are **com.teamcenter.rac.legacydataimport** and **com.teamcenter.rac.inprocessobjects**. From that custom User Interface has created in Teamcenter 12. We can browse the path by automatically by the help custom User Interface.

### 3.1 Configuration of Eclipse IDE with Teamcenter

In this customization, projects are built for molding the Teamcenter software as per the business need and the client need. The steps required for configuration of Eclipse Photon IDE with Teamcenter are given below in Figure [4]. For the plug-in project, I have written the source code in core Java language and also used the swing concept for the designing the user interface. I have browsed the file path in my local computer and exported data of Custom Item and Released Objects in MS Excel Format as per client's requirement.

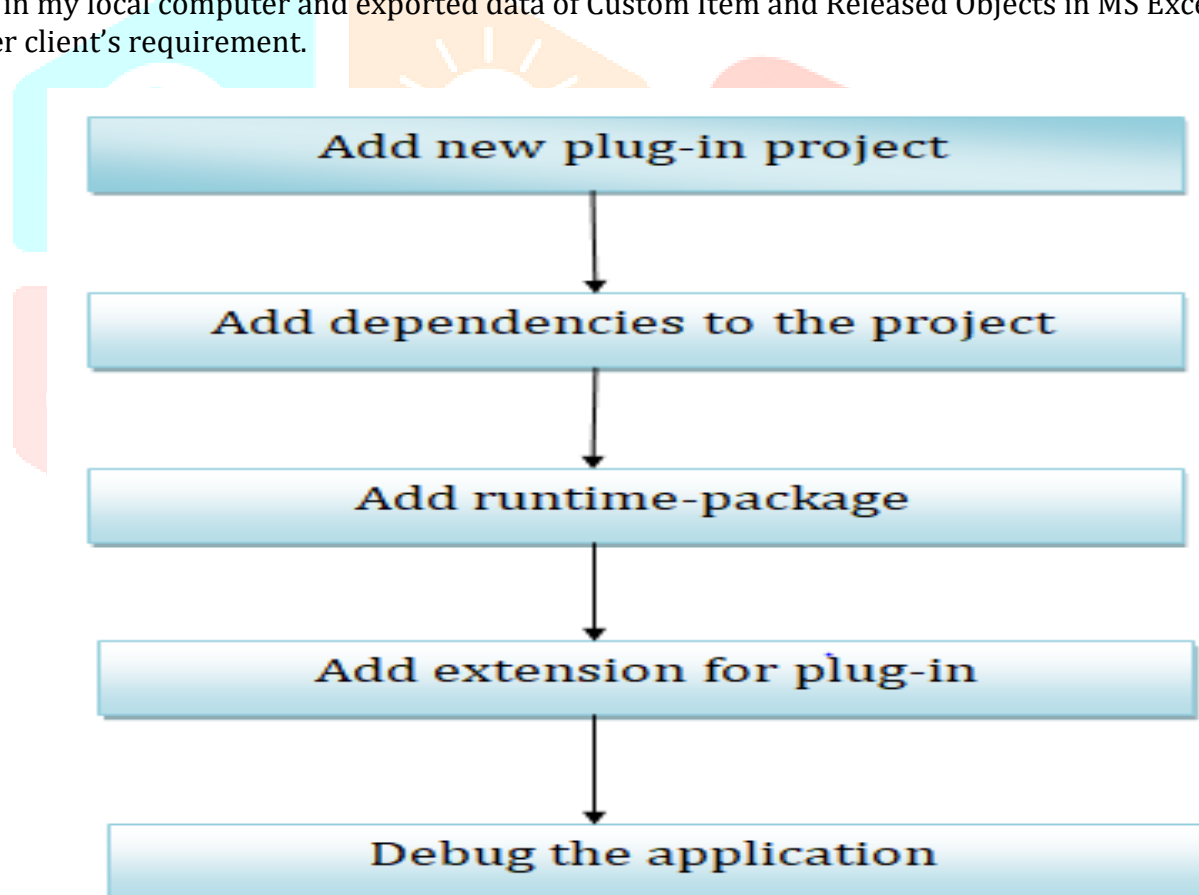
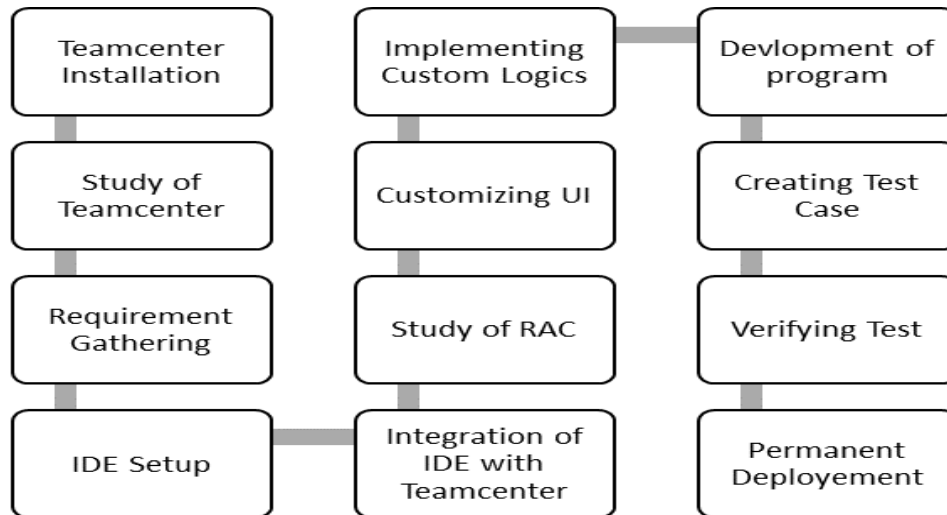


Fig -2: Configuration of Eclipse Photon with Teamcenter

### 3.2 Steps of Execution



**Fig-3:** Steps for Execution of Projects

### 3.3 Document representation of how to execute Legacy Data Import

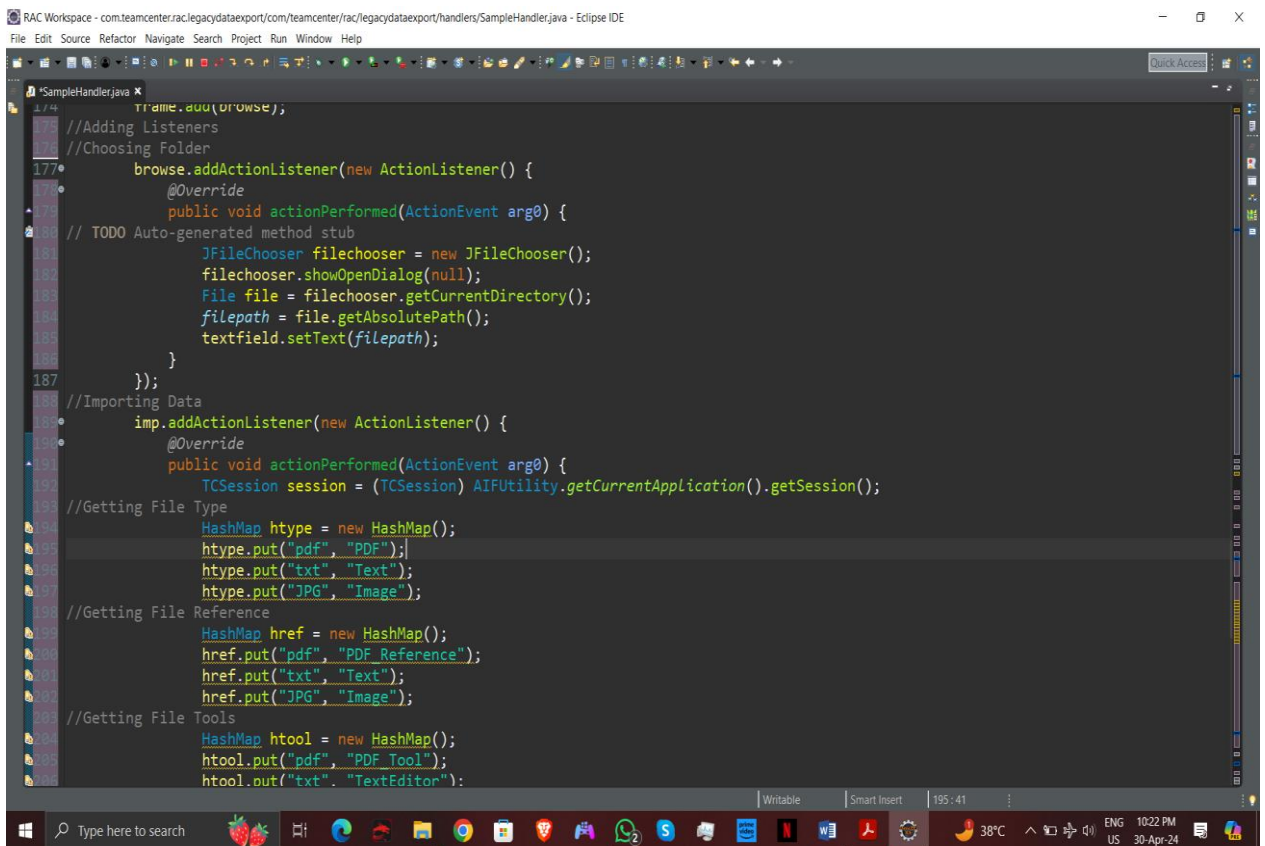
1. To create Eclipse based plug in project by using Eclipse Photon and write the code which is required for my data import activity. In that I have wrote the code for creating a file directory selection dialog upon client request using JSwing where client can select the directory where physical files are present.

```

RAC Workspace - com.teamcenter.rac.legacydataexport.com/teamcenter/rac/legacydataexport/handlers/SampleHandler.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
SampleHandler.java x
147 public class SampleHandler extends AbstractHandler {
148     static String filepath;
149
150     @Override
151     public Object execute(ExecutionEvent event) throws ExecutionException {
152         IWorkbenchWindow window = HandlerUtil.getActiveWorkbenchWindowChecked(event);
153         //Creating Components
154         JFrame frame = new JFrame("Legacy Data Import");
155         JLabel label = new JLabel("File: ");
156         final JTextField textfield = new JTextField();
157         JButton browse = new JButton("Browse");
158         JButton imp = new JButton("Import");
159         JButton cancel = new JButton("Cancel");
160         //Dimensioning Components
161         textfield.setBounds(50, 50, 350, 60);
162         browse.setBounds(450, 50, 100, 50);
163         imp.setBounds(70, 150, 100, 50);
164         cancel.setBounds(220, 150, 100, 50);
165         label.setBounds(25, 50, 50, 50);
166         frame.setSize(700, 700);
167         frame.setLayout(null);
168         frame.setVisible(true);
169         //Adding Components to Frame
170         frame.add(textfield);
171         frame.add(label);
172         frame.add(cancel);
173         frame.add(imp);
174         frame.add(browse);
175         //Adding Listeners
176         //Choosing Folder
177         browse.addActionListener(new ActionListener() {
178             @Override
  
```

**Fig -4:** Eclipse Photon IDE User Interface



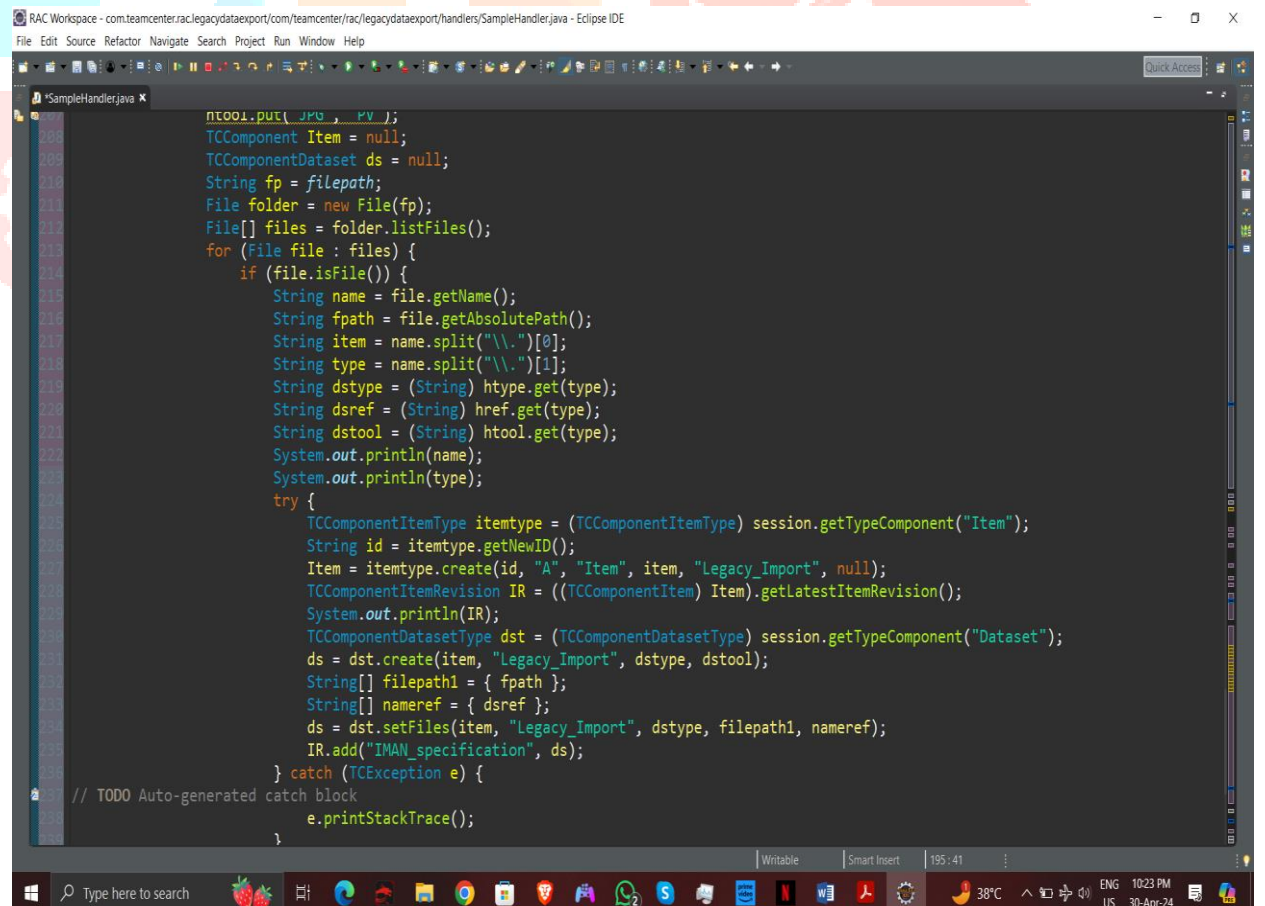


```

174 trame.add(browse);
175 //Adding Listeners
176 //Choosing Folder
177 browse.addActionListener(new ActionListener() {
178     @Override
179     public void actionPerformed(ActionEvent arg0) {
180         // TODO Auto-generated method stub
181         JFileChooser filechooser = new JFileChooser();
182         filechooser.showOpenDialog(null);
183         File file = filechooser.getCurrentDirectory();
184         filepath = file.getAbsolutePath();
185         textfield.setText(filepath);
186     }
187 });
188 //Importing Data
189 imp.addActionListener(new ActionListener() {
190     @Override
191     public void actionPerformed(ActionEvent arg0) {
192         TCSession session = (TCSession) AIFUtility.getCurrentApplication().getSession();
193         //Getting File Type
194         HashMap htype = new HashMap();
195         htype.put("pdf", "PDF");
196         htype.put("txt", "Text");
197         htype.put("JPG", "Image");
198         //Getting File Reference
199         HashMap href = new HashMap();
200         href.put("pdf", "PDF Reference");
201         href.put("txt", "Text");
202         href.put("JPG", "Image");
203         //Getting File Tools
204         HashMap htool = new HashMap();
205         htool.put("pdf", "PDF Tool");
206         htool.put("txt", "TextEditor");

```

Fig -5: Eclipse Photon IDE Java Swing Code



```

207 htool.put("JPG", "PV");
208 TCComponent Item = null;
209 TCComponentDataset ds = null;
210 String fp = filepath;
211 File folder = new File(fp);
212 File[] files = folder.listFiles();
213 for (File file : files) {
214     if (file.isFile()) {
215         String name = file.getName();
216         String fpath = file.getAbsolutePath();
217         String item = name.split("\\.")[0];
218         String type = name.split("\\.")[1];
219         String dstype = (String) htype.get(type);
220         String dsref = (String) href.get(type);
221         String dstool = (String) htool.get(type);
222         System.out.println(name);
223         System.out.println(type);
224         try {
225             TCComponentItemType itemtype = (TCComponentItemType) session.getTypeComponent("Item");
226             String id = itemtype.getNewID();
227             Item = itemtype.create(id, "A", "Item", item, "Legacy_Import", null);
228             TCComponentItemRevision IR = ((TCComponentItem) Item).getLatestItemRevision();
229             System.out.println(IR);
230             TCComponentDatasetType dst = (TCComponentDatasetType) session.getTypeComponent("Dataset");
231             ds = dst.create(item, "Legacy_Import", dstype, dstool);
232             String[] filepath1 = { fpath };
233             String[] nameref = { dsref };
234             ds = dst.setFiles(item, "Legacy_Import", dstype, filepath1, nameref);
235             IR.add("IMAN_specification", ds);
236         } catch (TCException e) {
237             // TODO Auto-generated catch block
238             e.printStackTrace();
239         }
240     }

```

Fig -6: Eclipse Photon IDE Legacy Data Import Code

2. We have seen the Custom menu in Teamcenter and this menu has added in Teamcenter.

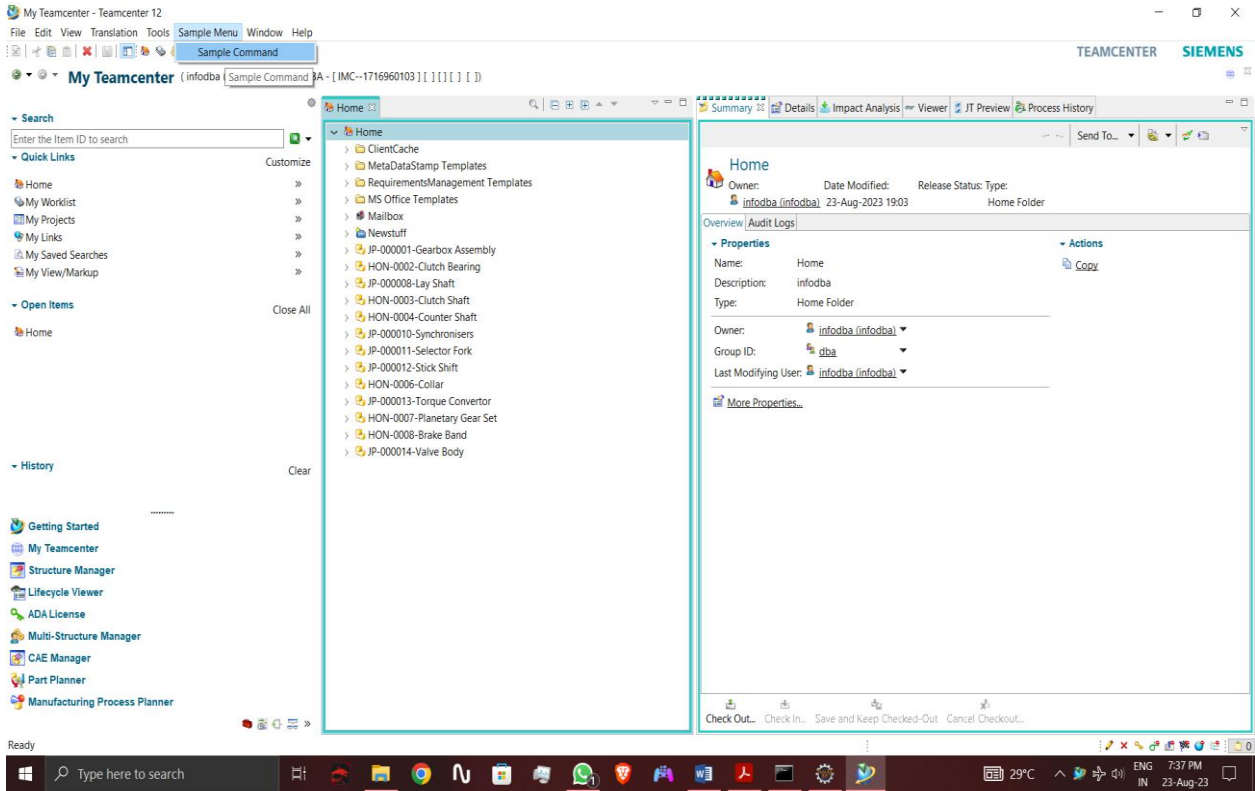


Fig -7: New Customized Menu Interface

3. By clicking the Custom Menu, custom swing UI get opened and there are two buttons one is Browse and second is Import. By using browse button user can select the file directory where he can select the directory in which physical files are present.

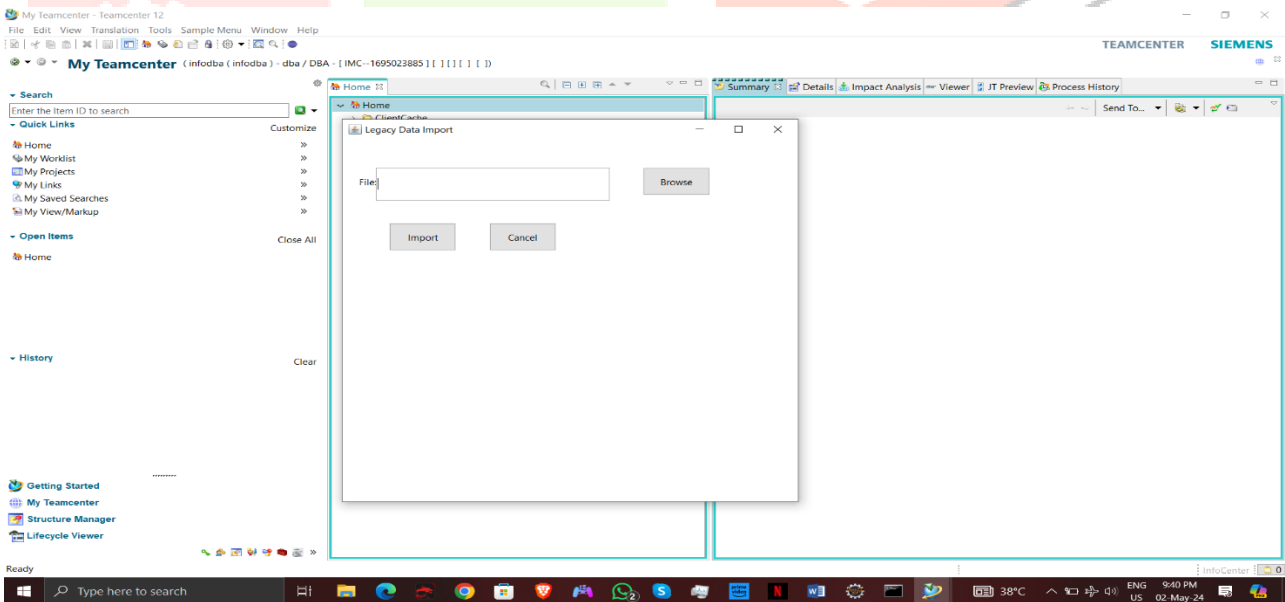


Fig -8: Custom Swing User Interface

4. After selecting the file directory when user clicks on Import button the files are imported into Teamcenter with same Item name, Revision name and Dataset name as physical file name.

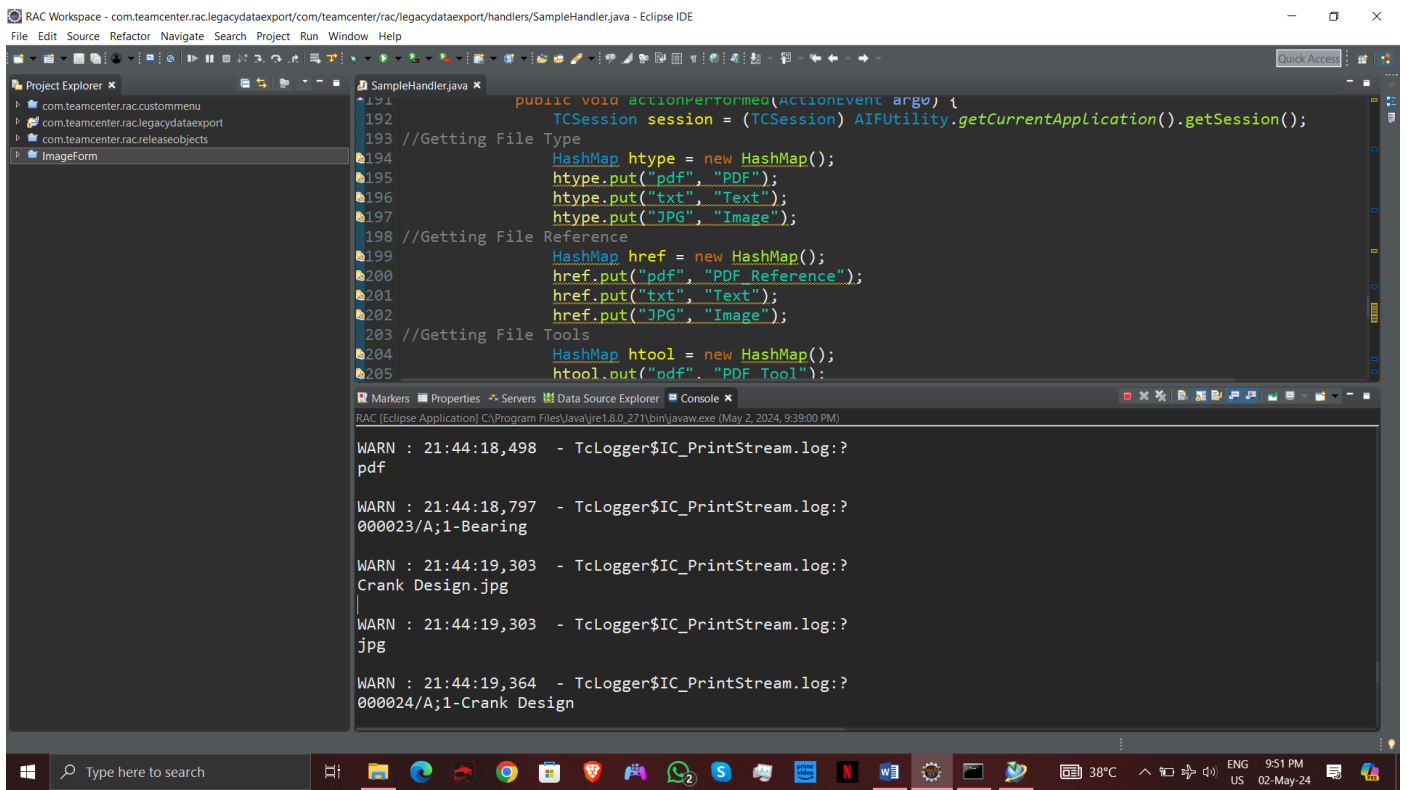


Fig -9: Eclipse Console Indicating Files are imported

5. For checking files are imported we have to search with respective Item ID’s of Items generated in console of Eclipse.

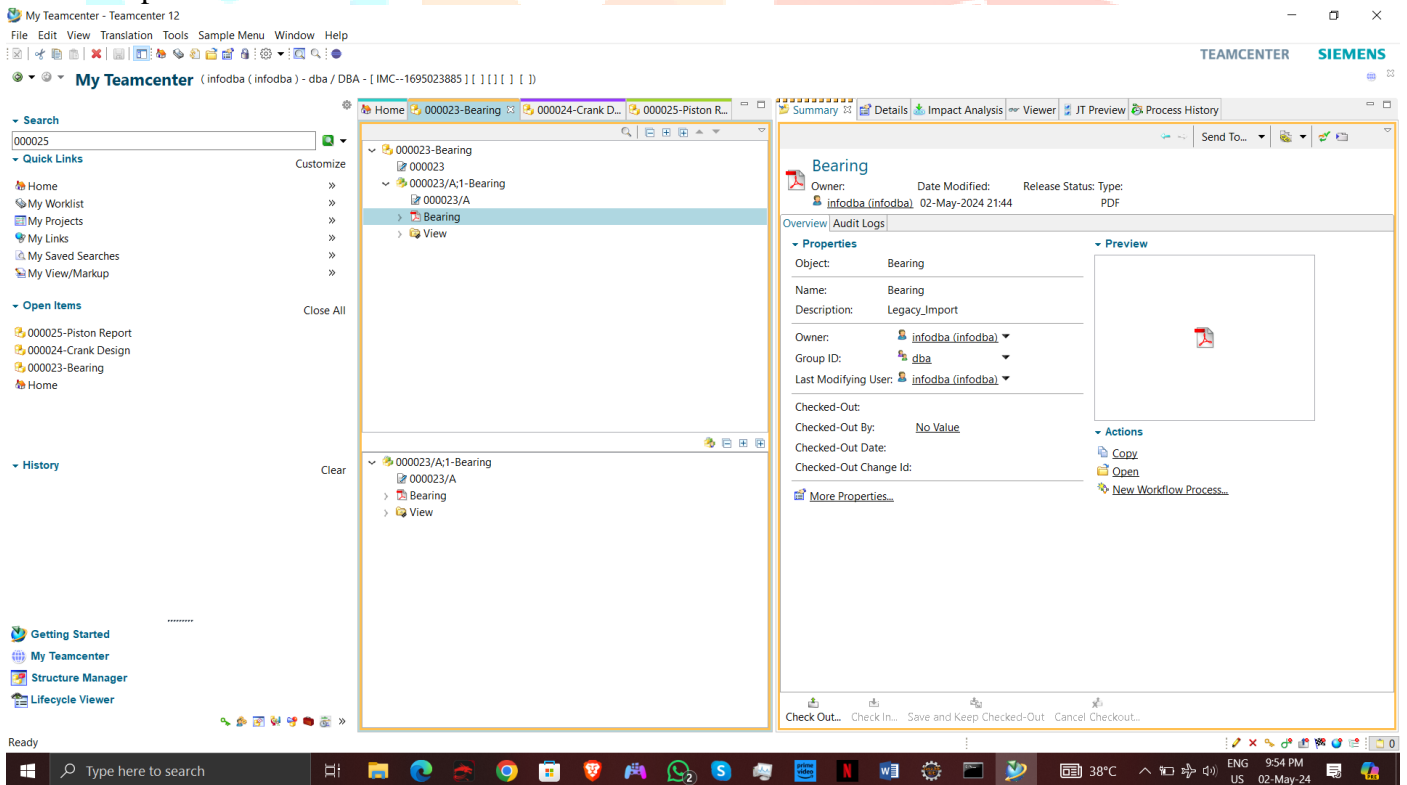


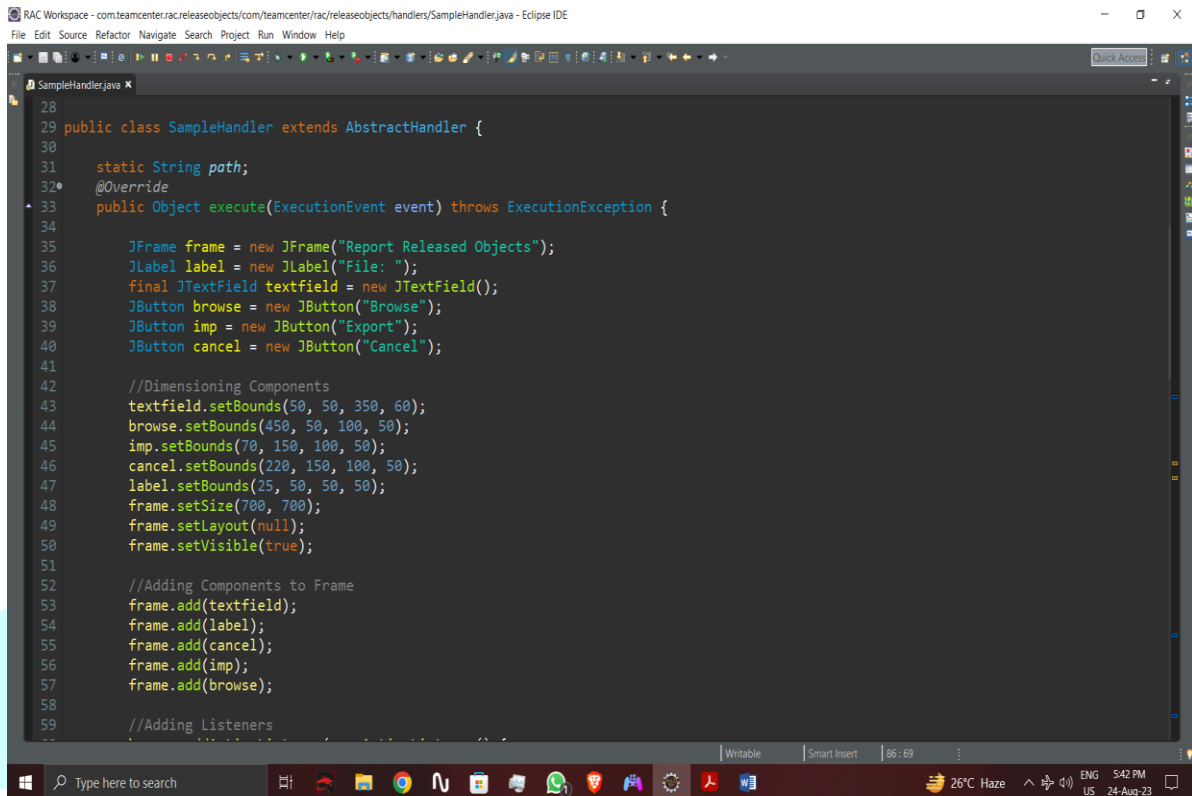
Fig -10: Data Imported in Teamcenter

6. By this way we can import the data of files using Rich Client Customization in Teamcenter rather using traditional method and it will save time for us.



### 3.4 Document representation of how to execute Workflow Inprocess Objects

1. To create Eclipse based plug in project by using Eclipse Photon and write the code which is required for my data export activity. In that I have wrote the code for creating a file directory selection dialog upon client request using JSwing where client can select the directory where report is to be generated.

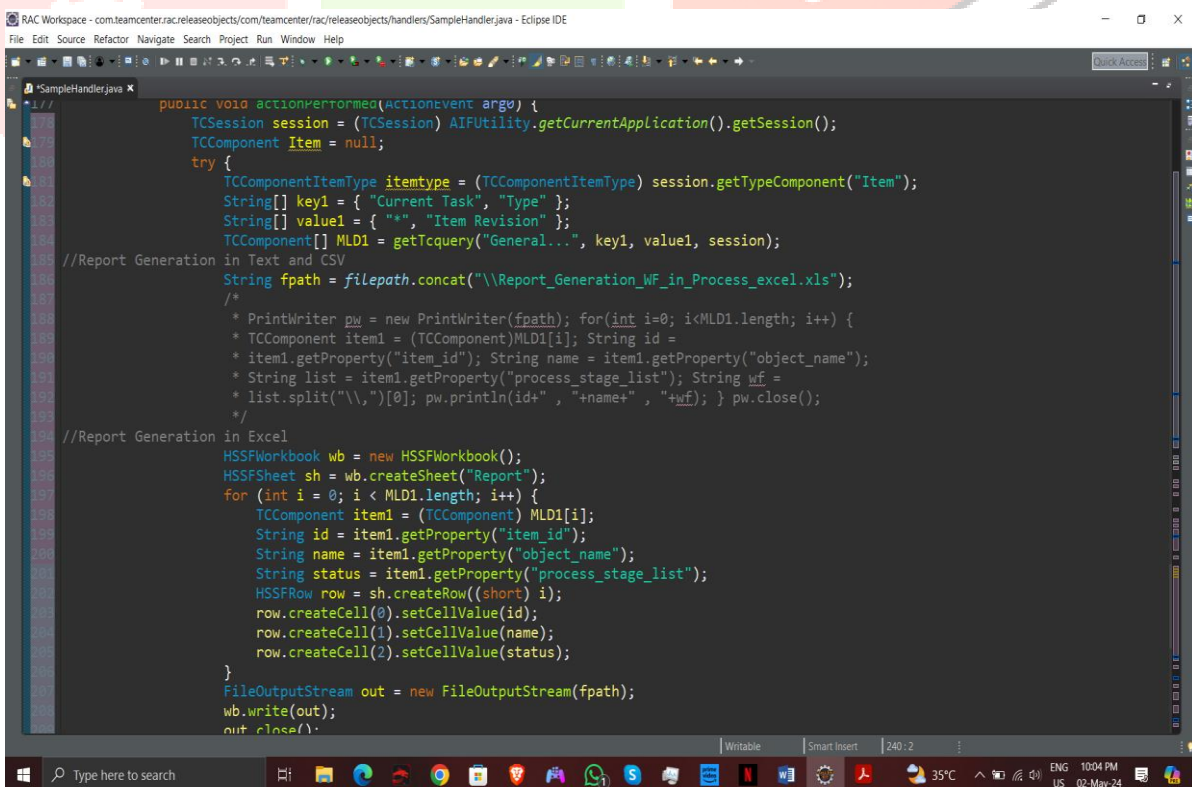


```

28
29 public class SampleHandler extends AbstractHandler {
30
31     static String path;
32     @Override
33     public Object execute(ExecutionEvent event) throws ExecutionException {
34
35         JFrame frame = new JFrame("Report Released Objects");
36         JLabel label = new JLabel("File: ");
37         JTextField textfield = new JTextField();
38         JButton browse = new JButton("Browse");
39         JButton imp = new JButton("Export");
40         JButton cancel = new JButton("Cancel");
41
42         //Dimensioning Components
43         textfield.setBounds(50, 50, 350, 60);
44         browse.setBounds(450, 50, 100, 50);
45         imp.setBounds(70, 150, 100, 50);
46         cancel.setBounds(220, 150, 100, 50);
47         label.setBounds(25, 50, 50, 50);
48         frame.setSize(700, 700);
49         frame.setLayout(null);
50         frame.setVisible(true);
51
52         //Adding Components to Frame
53         frame.add(textfield);
54         frame.add(label);
55         frame.add(cancel);
56         frame.add(imp);
57         frame.add(browse);
58
59         //Adding Listeners

```

Fig -11: Eclipse Code for Java Swing



```

177     public void actionPerformed(ActionEvent arg0) {
178         TCSession session = (TCSession) AIFUtility.getCurrentApplication().getSession();
179         TCComponent item = null;
180         try {
181             TCComponentItemType itemtype = (TCComponentItemType) session.getTypeComponent("Item");
182             String[] key1 = { "Current Task", "Type" };
183             String[] value1 = { "*", "Item Revision" };
184             TCComponent[] MLD1 = getTcquery("General...", key1, value1, session);
185             //Report Generation in Text and CSV
186             String fpath = filepath.concat("\\Report_Generation_WF_in_Process_excel.xls");
187             /*
188              * PrintWriter pw = new PrintWriter(fpath); for(int i=0; i<MLD1.length; i++) {
189              * TCComponent item1 = (TCComponent)MLD1[i]; String id =
190              * item1.getProperty("item_id"); String name = item1.getProperty("object_name");
191              * String list = item1.getProperty("process_stage_list"); String wf =
192              * list.split("\\,")[0]; pw.println(id+" "+name+" "+wf); } pw.close();
193              */
194             //Report Generation in Excel
195             HSSFWorkbook wb = new HSSFWorkbook();
196             HSSFSheet sh = wb.createSheet("Report");
197             for (int i = 0; i < MLD1.length; i++) {
198                 TCComponent item1 = (TCComponent) MLD1[i];
199                 String id = item1.getProperty("item_id");
200                 String name = item1.getProperty("object_name");
201                 String status = item1.getProperty("process_stage_list");
202                 HSSFRow row = sh.createRow((short) i);
203                 row.createCell(0).setCellValue(id);
204                 row.createCell(1).setCellValue(name);
205                 row.createCell(2).setCellValue(status);
206             }
207             FileOutputStream out = new FileOutputStream(fpath);
208             wb.write(out);
209             out.close();

```

Fig -12: Eclipse Code for Workflow Inprocess Objects

```
SampleHandler.java
//Report Generation in Excel
195 HSSFWorkbook wb = new HSSFWorkbook();
196 HSSFSheet sh = wb.createSheet("Report");
197 for (int i = 0; i < MLD1.length; i++) {
198     TCComponent item1 = (TCComponent) MLD1[i];
199     String id = item1.getProperty("item_id");
200     String name = item1.getProperty("object_name");
201     String status = item1.getProperty("process_stage_list");
202     HSSFRow row = sh.createRow((short) i);
203     row.createCell(0).setCellValue(id);
204     row.createCell(1).setCellValue(name);
205     row.createCell(2).setCellValue(status);
206 }
207 FileOutputStream out = new FileOutputStream(fpath);
208 wb.write(out);
209 out.close();
210 System.out.println("File is Created");
211 } catch (Exception e) {
212     // TODO: handle exception
213 }
214 frame.dispose();
215 }
216 });
217 cancel.addActionListener(new ActionListener() {
218     @Override
219     public void actionPerformed(ActionEvent arg0) {
220     // TODO Auto-generated method stub
221     frame.dispose();
222     }
223 });
224 return null;
225 }
```

Fig -13: Eclipse Code for Excel Write

2. We have seen the Custom menu in Teamcenter and this menu has added in Teamcenter.

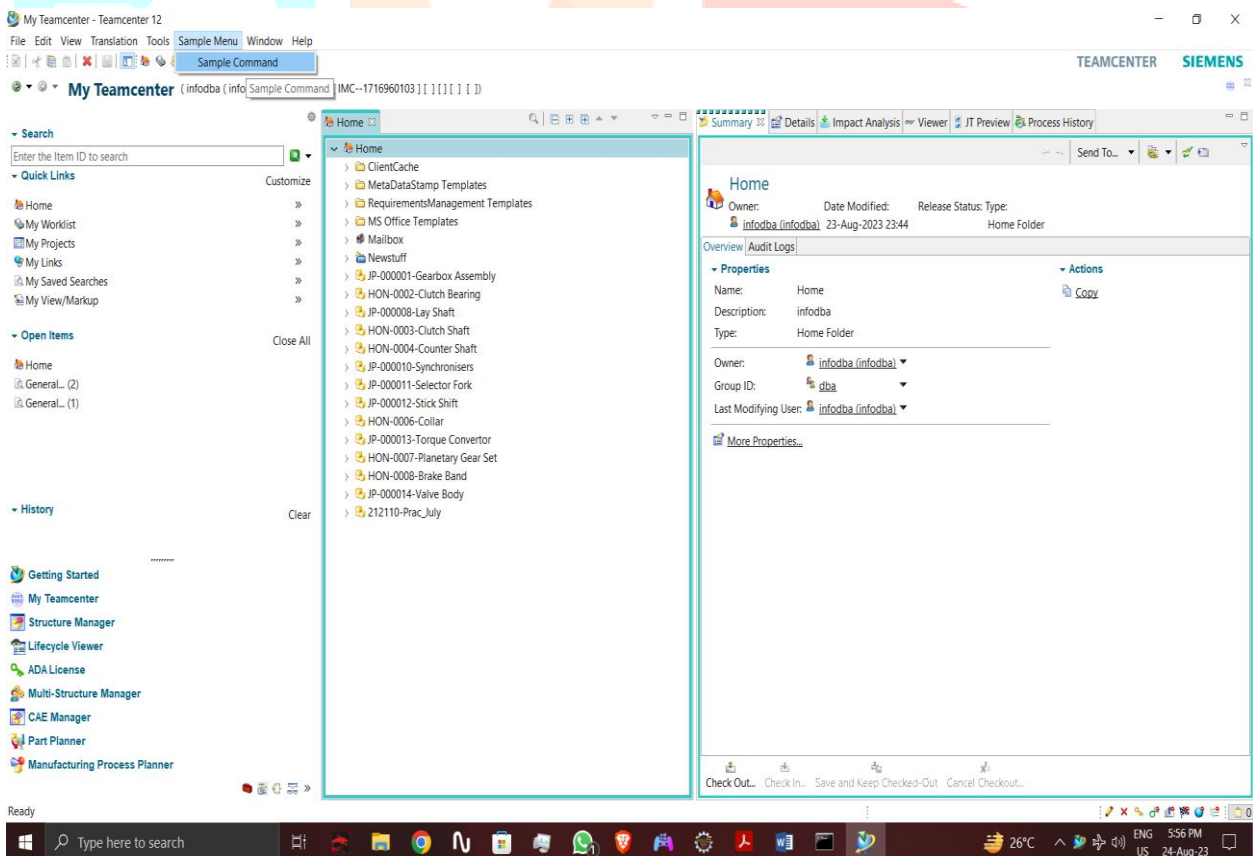
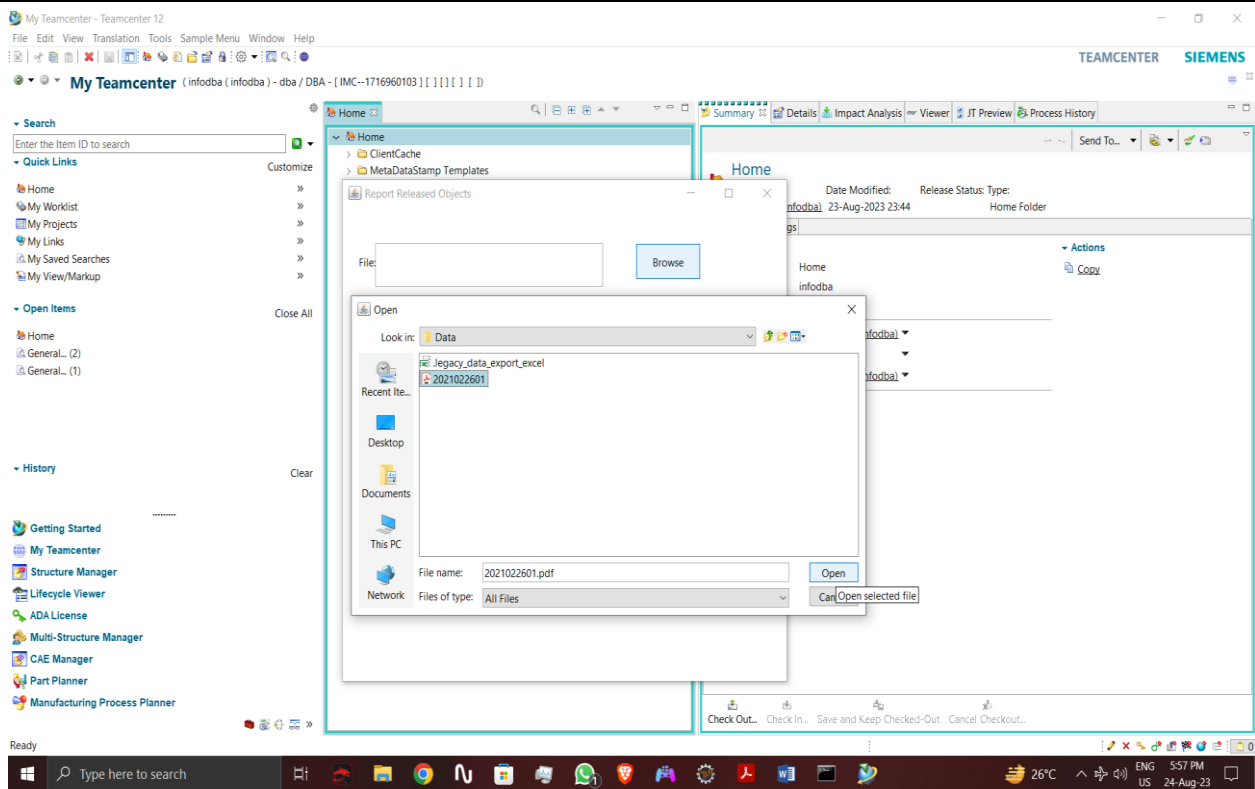


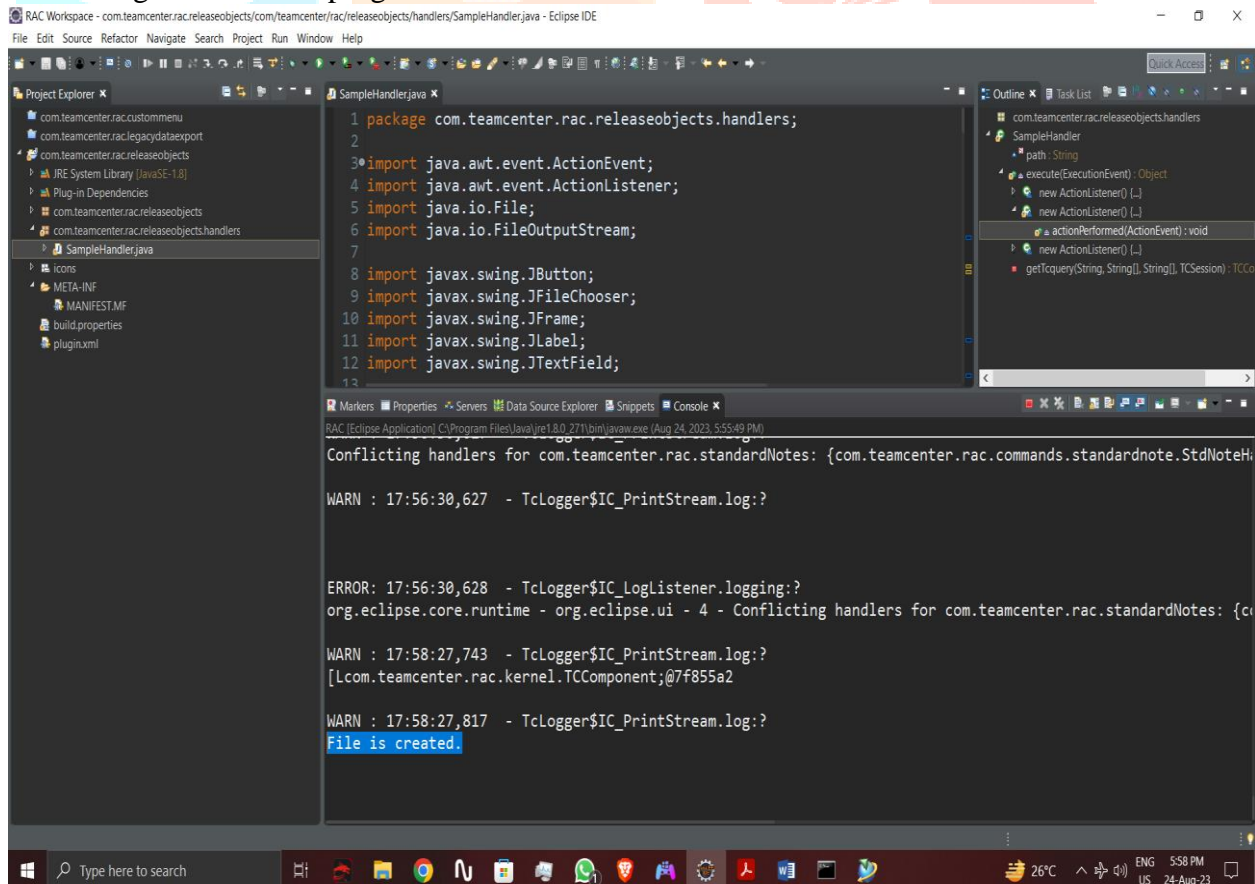
Fig -14: New Customized Menu Interface for Inprocess Objects

3. By clicking the Custom Menu, custom swing UI get opened and there are two buttons one is Browse and second is Export. By using browse button user can select the file directory where he wish to generate report.



**Fig -15: Custom Swing UI for Inprocess Objects**

- After selecting the file path when user clicks on Export button we have executed the default Query from Query Builder and by running this Query we got the list of Objects which are in process. Once Query is executed through code, the result is properly type cast and stored in array and data is written into MS Excel through Excel write program.



**Fig -16: Eclipse Console Indicating File is created**

- For checking file is created or not check the directory in which you have given the file path. By this way we can export the data of Workflow Inprocess Objects with Item ID, Item Name and Process Stage using Rich Client Customization in Teamcenter.

## IV. RESULTS AND DISCUSSION

### 4.1 Result for Legacy Data Import

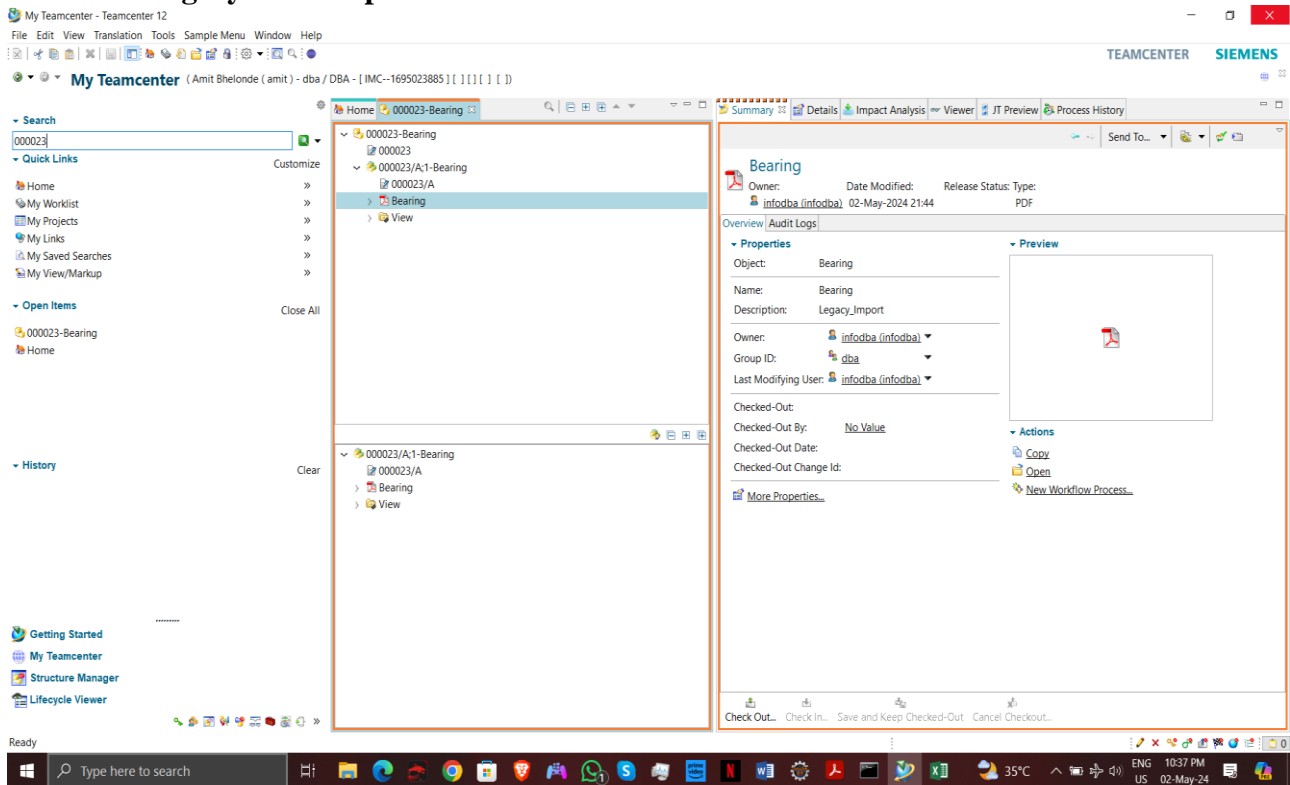


Fig -17: Imported files data in Teamcenter

As we have successfully compiled the code through Rich Client Customization, we have logged into Teamcenter and copied the Item ID's generated into Eclipse console. As we can see in the above fig. the Item has been created with same name with Revision and Dataset and physical file has been successfully imported into Teamcenter as per client's requirement. By this way I have carried this work successfully.

### 4.2 Result for Workflow Inprocess Objects

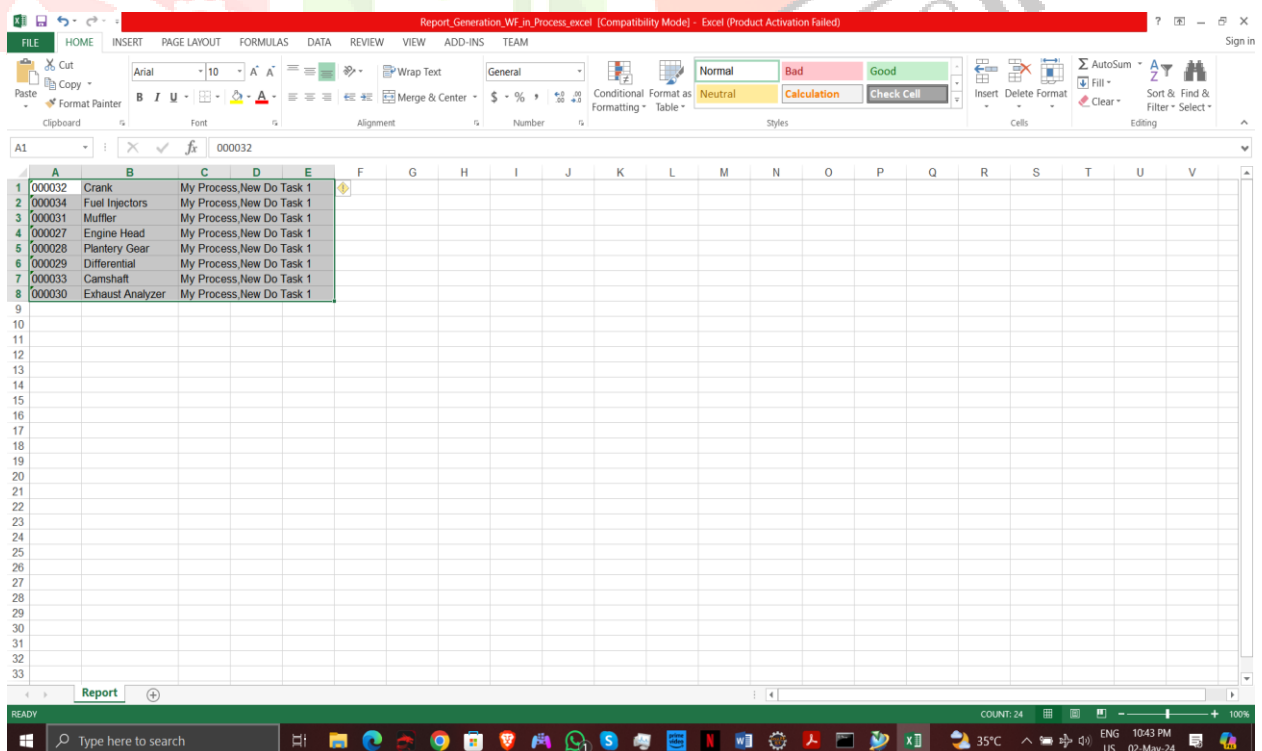


Fig -18: Exported Item Data in MS Excel format with Item ID, Name and Process Stage

As we have successfully compiled the code of Report Generation of Inprocess Objects, we have opened MS Excel to check whether the Released Objects Data with Item ID, Item Name and Process Stage have been exported or not. As we can see in the above fig. in first column Item ID's, second column Item



Name and third column Process Stage of Inprocess Objects have been successfully exported from Teamcenter as per client's requirement. By this way I have carried this work successfully.

## V. ACKNOWLEDGMENT

I am grateful to Mr. Sayyad Shafik for his valuable guidance and insightful feedback throughout the research process. I would like to extend my thanks to my company for giving me an opportunity to work on this project and all the friends who have helped me in gathering data related to this work.

## REFERENCES

- [1] Kurkin, Ondřej; Januška, Marlin (2010). "Product Life Cycle in Digital factory". Knowledge management and innovation: a business competitive edge perspective. Cairo: International Business Information Management Association (IBIMA): 1881–1886. ISBN 9780982148945.
- [2] Cunha, Luciano (20 July 2010). "Manufacturing Pioneers Reduce Costs By Integrating PLM & ERP". Onwindows.com. Archived from the original on 11 February 2017. Retrieved 7 February 2017.
- [3] Shrikant Pokale and Sawan Borul, "Client side customization for checking the user rights in Teamcenter-PLM" in International Journal of Applied Information Systems (IJ AIS) Volume 5, Issue 10, August-2013 ISSN 2249- 0868.
- [4] G. Shreenivasulu, "Eclipse Plug in based Teamcenter Customization" in International Journal of Advances in Computer Science and Technology Information Systems (IJACST) Volume 2, Issue 9, August-2013 ISSN 2320- 2602.
- [5] Teamcenter 12 Rich Architecture Client index file, Siemens Product Lifecycle Management Software Inc., 2018.
- [6] W. Schindel, —System interactions: Making the heart of systems more visible, || Proc. of INCOSE Great Lakes Regional Conference, 2013.
- [7] P. B. Hart, —A PLM implementation for aerospace systems engineering-conceptual rotorcraft design, || vol. M.S. thesis, Dept. Mechanical. Eng., Georgia Institute of Technology., Georgia, May 2009.
- [8] Shrikant Basarkod, by PLM Neutral Customization frame work, White paper, Geometric, June 2009.