

# **System and Software Architecture Description (SSAD)**

## **Data Mining of Digital Library Usage Data**

**Client:** Jewel Ward

### **Team #7**

Bo Lee

Genesan Kim

Maxim Krivokon

Vu Nguyen

# Version History

Date	Author	Version	Changes made
9/28/04	Hsiao-Han Huang Pei-Han Li	1.0	<ul style="list-style-type: none"><li>• Early Section</li></ul>
10/08/04	Hsiao-Han Huang Pei-Han Li	1.1	<ul style="list-style-type: none"><li>• Correction and add diagrams and tables</li></ul>
10/14/04	Hsiao-Han Huang Pei-Han Li	1.2	<ul style="list-style-type: none"><li>• Finish the section 3</li></ul>
10/24/04	Hsiao-Han Huang Pei-Han Li	2.0	<ul style="list-style-type: none"><li>• Correction and add 3.5</li></ul>
11/17/04	Hsiao-Han Huang Pei-Han Li	3.1	<ul style="list-style-type: none"><li>• Correct section 2 section 3 and add section4</li></ul>
12/6/04	Hsiao-Han Huang Pei-Han Li	4.0	<ul style="list-style-type: none"><li>• Complete SSAD at LCA stage</li></ul>
02/09/05	Maxim Krivokon	5.0	<ul style="list-style-type: none"><li>• Rewritten section 3</li></ul>
02/25/05	Maxim Krivokon	7.0	<ul style="list-style-type: none"><li>• Rewritten section 4 – see 1.4 Change Summary</li></ul>

# Table of Contents

<b>SYSTEM AND SOFTWARE ARCHITECTURE DESCRIPTION (SSAD)</b> .....	<b>1</b>
<b>VERSION HISTORY</b> .....	<b>2</b>
<b>TABLE OF CONTENTS</b> .....	<b>3</b>
<b>TABLE OF TABLES</b> .....	<b>4</b>
<b>TABLE OF FIGURES</b> .....	<b>5</b>
1. Introduction.....	9
1.1 Purpose of the SSAD Document.....	9
1.2 Standards and Conventions.....	9
1.3 References.....	9
1.4 Change Summary.....	11
2. System Analysis.....	12
2.1 Structure.....	12
2.2 Artifacts & Information .....	13
2.3 Behavior.....	14
2.4 L.O.S. Goals.....	24
2.5 Rules .....	24
3. Architecture Design & Analysis .....	25
3.1 Structure.....	25
3.2 Analysis Classes.....	73
3.3 Behavior.....	75
3.4 L.O.S Projected.....	84
3.5 Architectural Styles, Patterns & Frameworks.....	85
4. Implementation Design.....	86
4.1 Structure.....	86
4.2 Behavior.....	115
4.3 L.O.S. projected .....	123
4.4 Patterns & Frameworks.....	124
4.5 Project Artifacts .....	125
5. Glossary for System Analysis and Design.....	126
6. Appendices.....	126

# Table of Tables

<i>Table 1 Use-Case Description for Import usage data</i> .....	16
<i>Table 2 Use-Case Description for Remove usage data</i> .....	17
<i>Table 3 Use-Case description for Generate analysis report</i> .....	19
<i>Table 4 Use-Case Description for Remove analysis report</i> .....	20
<i>Table 5 Use-Case Description for Open analysis report</i> .....	21
<i>Table 6 Use-Case description for Browse analysis report</i> .....	23
<i>Table 7 Available Capabilities &amp; Processes in Operational Mode</i> .....	24
<i>Table 8 L.O.S Goals</i> .....	24
<i>Table 9 LOS goals for Controller component (LR-1)</i> .....	69
<i>Table 10 LOS goals for Controller component (LR-2)</i> .....	69
<i>Table 11 LOS goals for Controller component (LR-4)</i> .....	69
<i>Table 12 LOS goals for Controller component (LR-5)</i> .....	70
<i>Table 13 L.O.S projected</i> .....	85
<i>Table 14 Architectural Styles, Patterns &amp; Frameworks</i> .....	85
<i>Table 15 Hardware Classifier Implementation</i> .....	86
<i>Table 28L.O.S projected</i> .....	124
<i>Table 29 Architectural Styles, Patterns &amp; Frameworks</i> .....	124

# Table of Figures

<i>Figure 1 System Static Structure Diagram .....</i>	<i>12</i>
<i>Figure 2 System Collaboration Diagram.....</i>	<i>13</i>
<i>Figure 3 Artifact model .....</i>	<i>14</i>
<i>Figure 4 System Processes .....</i>	<i>15</i>
<i>Figure 5 Activity diagram for Import usage data.....</i>	<i>16</i>
<i>Figure 6 Activity diagram for Remove usage data .....</i>	<i>18</i>
<i>Figure 7 Activity diagram for Generate analysis report .....</i>	<i>19</i>
<i>Figure 8 Activity diagram for Remove analysis report .....</i>	<i>20</i>
<i>Figure 9 Activity diagram for Visualize analysis report .....</i>	<i>22</i>
<i>Figure 10 Activity diagram for Browse analysis report.....</i>	<i>23</i>
<i>Figure 11 Topology of the system .....</i>	<i>25</i>
<i>Figure 12 Hardware Classifier Model.....</i>	<i>27</i>
<i>Figure 13 View layer component diagram.....</i>	<i>28</i>
<i>Figure 14 Controller layer component diagram.....</i>	<i>28</i>
<i>Figure 15 Model software component diagram.....</i>	<i>29</i>
<i>Figure 16 Deployment Model.....</i>	<i>30</i>
<i>Figure 17 GUI Interface.....</i>	<i>32</i>
<i>Figure 18 GUI processes use-case diagram.....</i>	<i>35</i>
<i>Figure 19 View logs process Activity diagram .....</i>	<i>36</i>
<i>Figure 20 Invoke importLogs Activity Diagram.....</i>	<i>38</i>

<i>Figure 21 Invoke removeLogs Activity Diagram</i> .....	39
<i>Figure 22 View Relation Sets Activity Diagram</i> .....	40
<i>Figure 23 Invoke makeRelationSet Activity Diagram</i> .....	42
<i>Figure 24 Invoke removeRelationSet Activity Diagram</i> .....	43
<i>Figure 25 View Object Trees Activity Diagram</i> .....	44
<i>Figure 26 Invoke makeObjectTree Activity Diagram</i> .....	46
<i>Figure 27 Invoke removeObjectTree</i> .....	47
<i>Figure 28 Open Object Tree Activity Diagram</i> .....	48
<i>Figure 29 LOS Goals for GUI Component Classifier</i> .....	49
<i>Figure 30 Visualizer interface</i> .....	49
<i>Figure 31 Visualizer processes Use-Case diagram</i> .....	50
<i>Figure 32 Visualize Object Tree Activity Diagram</i> .....	51
<i>Figure 33 Browse Object Tree Activity Diagram</i> .....	53
<i>Figure 34 LOS Goals for Visualizer component</i> .....	54
<i>Figure 35 Controller interface</i> .....	55
<i>Figure 36 Controller Processes Use-Case Diagram</i> .....	57
<i>Figure 37 import Logs Activity Diagram</i> .....	58
<i>Figure 38 List Logs process Activity Diagram</i> .....	59
<i>Figure 39 Remove Logs process Activity Diagram</i> .....	61
<i>Figure 40 Make Relation Set Activity Diagram</i> .....	62
<i>Figure 41 List Relation Sets Activity Diagram</i> .....	63
<i>Figure 42 Remove Relation Set Activity Diagram</i> .....	64

<i>Figure 43 Make Object Tree Activity Diagram</i> .....	66
<i>Figure 44 List Object Trees process Activity Diagram</i> .....	67
<i>Figure 45 Remove Object Tree Activity Diagram</i> .....	68
<i>Figure 46 Information Classes</i> .....	73
<i>Figure 47 System process implementation</i> .....	75
<i>Figure 48 Import Logs Implementation</i> .....	76
<i>Figure 49 Remove Logs Implementation</i> .....	78
<i>Figure 50 Generate Relation Set Implementation</i> .....	79
<i>Figure 51 Remove Relation Set Implementation</i> .....	80
<i>Figure 52 Generate Object Tree Implementation</i> .....	81
<i>Figure 53 Remove Object Tree Implementation</i> .....	82
<i>Figure 54 Browse Object Tree Implementation</i> .....	84
<i>Figure 55 View layer Implementation</i> .....	87
<i>Figure 56 Controller layer Implementation</i> .....	87
<i>Figure 57 Model layer Implementation</i> .....	88
<i>Figure 58 Deployment Model</i> .....	88
<i>Figure 59 Open Motif GUI Class Diagram</i> .....	91
<i>Figure 60 Openotif GUI Collaboration Diagram</i> .....	92
<i>Figure 61 H3Viewer Class Diagram</i> .....	94
<i>Figure 62 H3Viewer Collaboration Diagram</i> .....	94
<i>Figure 63 C++ Controller Class Diagram</i> .....	96
<i>Figure 64 C++ Controller Collaboration Diagram</i> .....	97

<i>Figure 65 MainForm Interface .....</i>	<i>101</i>
<i>Figure 66 NewRelationSetForm Interface .....</i>	<i>103</i>
<i>Figure 67 NewObjectTreeForm Interface.....</i>	<i>106</i>
<i>Figure 68 H3ViewerFrame Interface.....</i>	<i>107</i>
<i>Figure 69 ObjectInfoFrame Interface.....</i>	<i>109</i>
<i>Figure 70 Clustering Object Tree Insterface .....</i>	<i>113</i>
<i>Figure 71 System Processes Realization.....</i>	<i>115</i>
<i>Figure 72 Add Logs Realization Sequence Diagram.....</i>	<i>116</i>
<i>Figure 73 Remove Logs realization sequence diagram.....</i>	<i>117</i>
<i>Figure 74 Make Relation Set Realization .....</i>	<i>118</i>
<i>Figure 75 Remove Relation Set Realization.....</i>	<i>119</i>
<i>Figure 76 Make Object Tree Realization.....</i>	<i>120</i>
<i>Figure 77 Remove Object Tree Realization .....</i>	<i>121</i>
<i>Figure 78 Open Object Tree Realization .....</i>	<i>122</i>
<i>Figure 79 Browse Object Tree Realization.....</i>	<i>123</i>
<i>Figure 80 Directory structure for Project.....</i>	<i>125</i>
<i>Figure 81 Src directory content .....</i>	<i>125</i>
<i>Figure 82 Include directory content.....</i>	<i>125</i>
<i>Figure 83 Bin directory content .....</i>	<i>126</i>

# 1. Introduction

## 1.1 Purpose of the SSAD Document

This document is a refinement of architecture description compiled during the inception phase. The purpose of this document is to bring all the necessary details into architecture model and make it implementation specific. This document provides clear and sufficient description of system's structure and design and thus makes transition to implementation phase more efficient.

## 1.2 Standards and Conventions

- Standard
  - MBASE Guideline for 577b version 2.4.2
- Notation
  - UML: version 1.4
- Naming Conventions
  - Components and object are Nouns.
  - Behaviors and Operations are Verbs.

## 1.3 References

MBASE Guidelines version 2.4.2

[http://sunset.usc.edu/classes/cs577a\\_2004/guidelines/MBASE\\_Guidelines\\_v2.4.1.pdf](http://sunset.usc.edu/classes/cs577a_2004/guidelines/MBASE_Guidelines_v2.4.1.pdf)

MBASE Electronic Process Guide

<http://cse.usc.edu/research/MBASE/EPG>

USC Information Services Division

<http://www.usc.edu/isd/about/about.html>

Fall 2004 CS 577a Project #7 Description

[http://sunset.usc.edu/classes/cs577a\\_2004/projects/description/project7.htm](http://sunset.usc.edu/classes/cs577a_2004/projects/description/project7.htm)

UML Guidelines

[http://sunset.usc.edu/classes/cs577a\\_2003/coursenotes/ep/Introduction\\_to\\_UML.pdf](http://sunset.usc.edu/classes/cs577a_2003/coursenotes/ep/Introduction_to_UML.pdf)

CS577a – Software Engineering I website

[http://sunset.usc.edu/classes/cs577a\\_2004/](http://sunset.usc.edu/classes/cs577a_2004/)

Fall 2003 CS 577a Project #8 LCO portion of OCD

[http://ebase.usc.edu/eservices/cs577a\\_2003/team08a/LCO/OCD\\_LCO\\_F03a\\_T08.pdf](http://ebase.usc.edu/eservices/cs577a_2003/team08a/LCO/OCD_LCO_F03a_T08.pdf)

Fall 2003 CS 577a Project #8 LCO portion of SSAD

[http://www-scf.usc.edu/%7Ecsci577/www/team8a/LCO/SSAD\\_LCO\\_F03a\\_T08.doc](http://www-scf.usc.edu/%7Ecsci577/www/team8a/LCO/SSAD_LCO_F03a_T08.doc)

Fall 2003 CS 577a Project #22 LCO portion of SSAD

[http://www-scf.usc.edu/%7Ecsci577/www/team22a/LCO/SSAD\\_LCO\\_F03a\\_T22.do](http://www-scf.usc.edu/%7Ecsci577/www/team22a/LCO/SSAD_LCO_F03a_T22.do)

[c](#)

Fall 2003 CS 577a Project #15 LCO portion of SSAD

[http://www-scf.usc.edu/%7Ecsci577/www/team15a/LCO/SSAD/SSAD\\_LCO\\_F03a\\_T15.doc](http://www-scf.usc.edu/%7Ecsci577/www/team15a/LCO/SSAD/SSAD_LCO_F03a_T15.doc)

Fall 2004 CS 577a Project #7 OCD Document

## 1.4 Change Summary

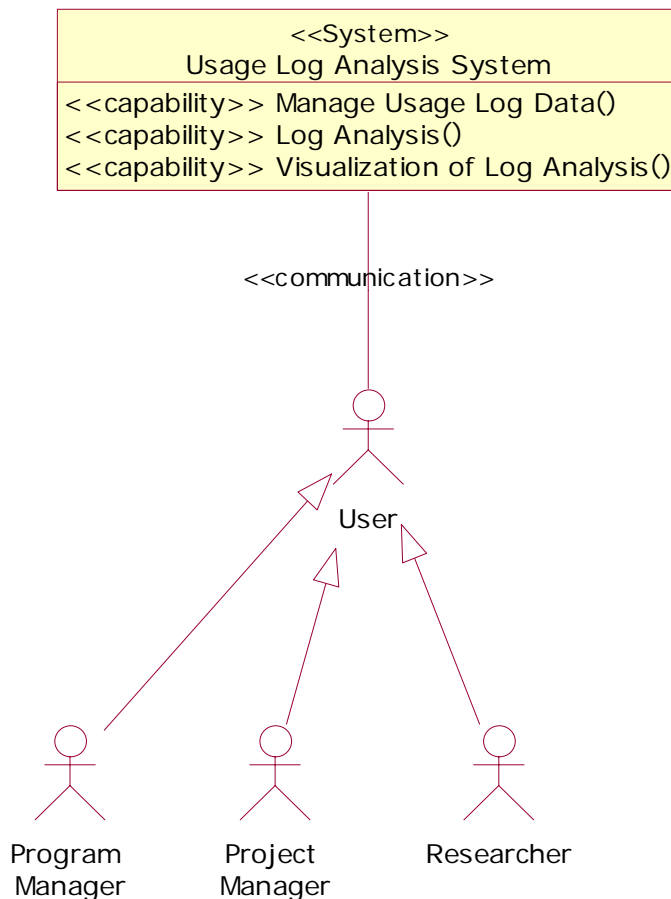
No	Change	Rationale	Sections Affected
1	removed description for business workers	system has one generic user	2.1
2	introduced new system artifacts: Item usage, Item tree, Item graph	consistency with system processes which operate on these artifacts	2.2, 2.3.1
3	Replaced 3 system processes with 6 new processes that map to System Capabilities defined in OCD and cover behavior of all System components	Consistency with OCD  Cover behaviors of all system components	2.3, 3.3, 4.3
4	updated three layered architecture to Model View Controller is an established design pattern.	Changed naming of three layers. Previously used – Process – which is more applicable to business applications  separation of concerns, previous experience	System Topology
5	Separated component into three models for each layer  Replaced 4 components by 2 in View Layer, 4 in Controller  Defined interfaces for each component		System Components
6	Section 4 was rewritten completely.	Inconsistencies in the old document. Lack of architectural detail	3, 4

## 2. System Analysis

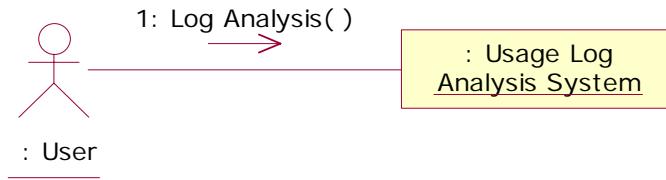
This section refines the proposed system in terms of its interactions with its users and artifacts it operates on as it was defined in OCD 4 “Proposed system”.

### 2.1 Structure

We describe the actors who interact with the system when it is operational. The proposed system does not have different modes of behavior based on which business or outside actor is interacting with it. Therefore the system will have one generic user and will provide all of its capabilities in one operational mode.



**Figure 1 System Static Structure Diagram**



**Figure 2 System Collaboration Diagram**

Figure 3 shows one possible configuration of system usage – the user invokes one of the system’s capabilities – “Log Analysis”.

## 2.1.1 System

The proposed system is a tool that allows visualization and analysis of digital library usage data. Provided with input log files the system generates and visualizes digital library item collection structure that reflects similarity relationships between items that are viewed together most often. In summary the system provides the following major capabilities:

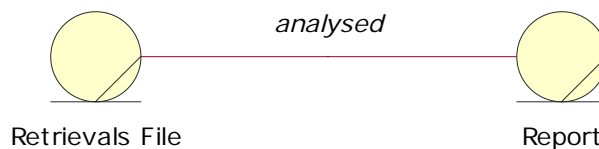
- Manage usage log data
- Log Analysis
- Visualization of log analysis result\

## 2.1.2 User

The user of the proposed system is a generic actor that is expected to have experience using other software and being familiar with common computer interface concepts such like input forms, action buttons etc. The user does not have any attributes used by the system and is not required to provide any services.

## 2.2 Artifacts & Information

The proposed system takes a Retrievals File as input data and analyses it to produce Report that contains information on item relations, their hierarchy and usage information for each item.



**Figure 3 Artifact model**

## **2.2.1 Retrievals file**

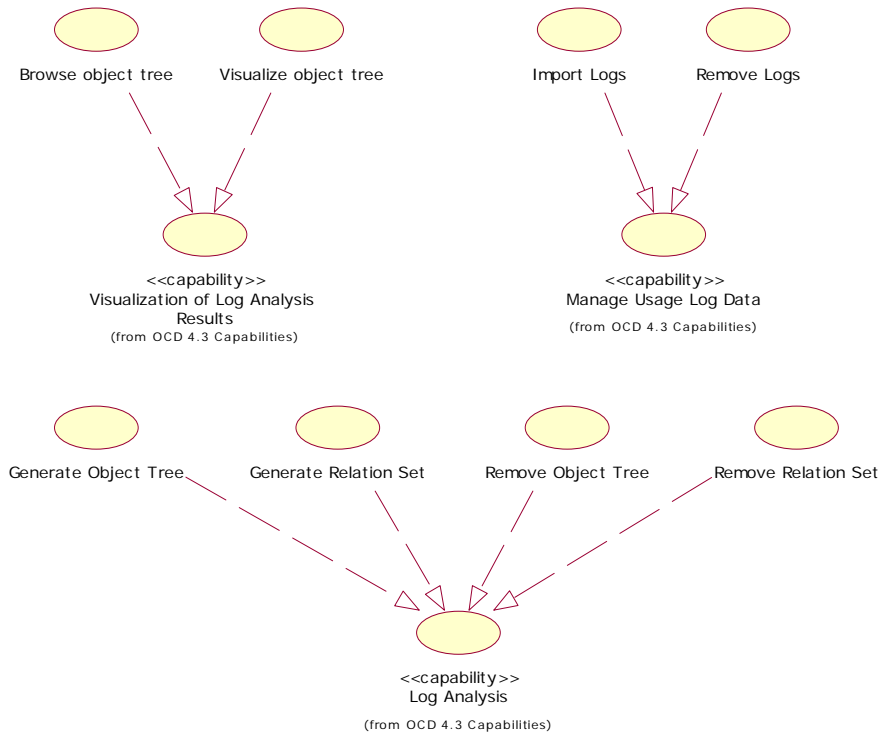
The Retrievals file is a text file with each line representing item retrieval from DA collection. This artifact is necessary as input data for the system's "Log Analysis" capability. "Manage log data" capability allows the user to specify location of the input log file and import the usage data into the system's database. The system tries to import the following data fields for each retrieval: time of retrieval, user id, item id, session id.

## **2.2.2 Report**

The "Log Analysis" capability mines the imported retrieval data for object relationships based on their co-retrievals and then orders those relations in hierarchical manner. Also usage information for each object is collected and stored.

## **2.3 Behavior**

This section shows the processes that realize the each of the system's capability defined in OCD section 4.3.



**Figure 4 System Processes**

As seen from the above diagram there are six processes that implement three major capabilities of the system. These processes will be described in more detail in the following sections.

## 2.3.1 Processes

### 2.3.1.1 Import usage data (UC-01)

<b>Identifier</b>	UC-01
<b>Use-Case Name</b>	Import Usage Data
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how usage data is imported
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None

<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	After user specifies source log file and initiates importing, system parses retrieval records from the file and stores them to the database.
<b>User Interface</b>	User is provided with standard file selection-browsing menu to pick the source log file. User is provided with a “Import” button to initiate importing of data.
<b>Pre-conditions</b>	Source log file is available from the local disk.
<b>Post-conditions</b>	System’s database contains representation of all well-formatted retrieval records from the source file
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

Table 1 Use-Case Description for Import usage data

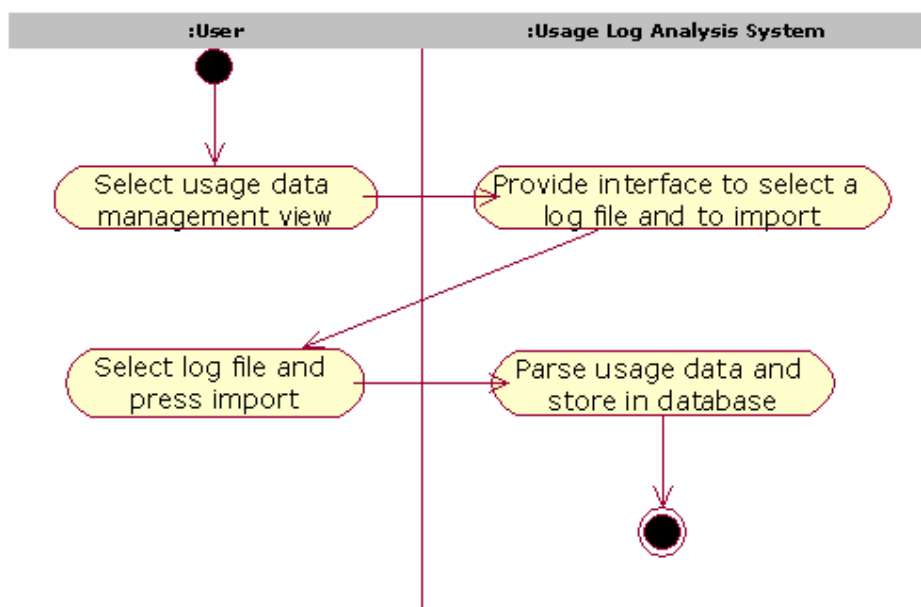


Figure 5 Activity diagram for Import usage data

### 2.3.1.2 Remove usage data (UC-02)

<b>Identifier</b>	UC-02
-------------------	-------

<b>Use-Case Name</b>	Remove usage data
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user can remove old or otherwise outdated usage information.
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-2 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	User is presented with a list of filenames from which usage data was imported previously. After user selected a specific file and pressed delete button, the system removes all retrieval records that were imported from that file.
<b>User Interface</b>	Already imported files are presented in a scrollable selectable list.
<b>Pre-conditions</b>	Usage data was previously imported from one or more files
<b>Post-conditions</b>	System's database does not contain any retrieval record that was associated with the selected log file
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

Table 2 Use-Case Description for Remove usage data

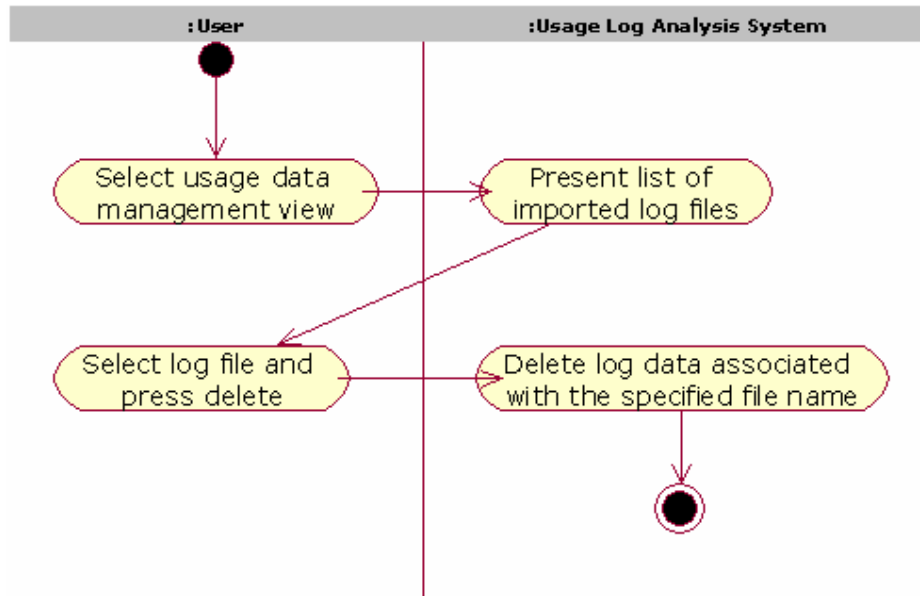


Figure 6 Activity diagram for Remove usage data

### 2.3.1.3 Generate analysis report (UC-03)

<b>Identifier</b>	UC-03
<b>Use-Case Name</b>	Generate analysis report
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user can generate analysis report
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-02 (OCD 4.3.2)
<b>Requirements</b>	SR-3,4,5 (SSRD 3.2.1), IR-2 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	In order to generate a new analysis report user has to select date range of usage data to be used and parameters for analysis algorithm. After the user initiates report generation, system generated item relationship graph from the specified usage data and collects usage statistic for each item. Based on the generated item relationship graph system produces

	collection structure tree by applying multi-level graph clustering and centrality analysis.
<b>User Interface</b>	User is provided with a form for inputting algorithm parameters.
<b>Pre-conditions</b>	Usage data was previously imported from one or more log files.
<b>Post-conditions</b>	System database contains representation of collection structure tree and usage statistics for each item in that tree.
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

Table 3 Use-Case description for Generate analysis report

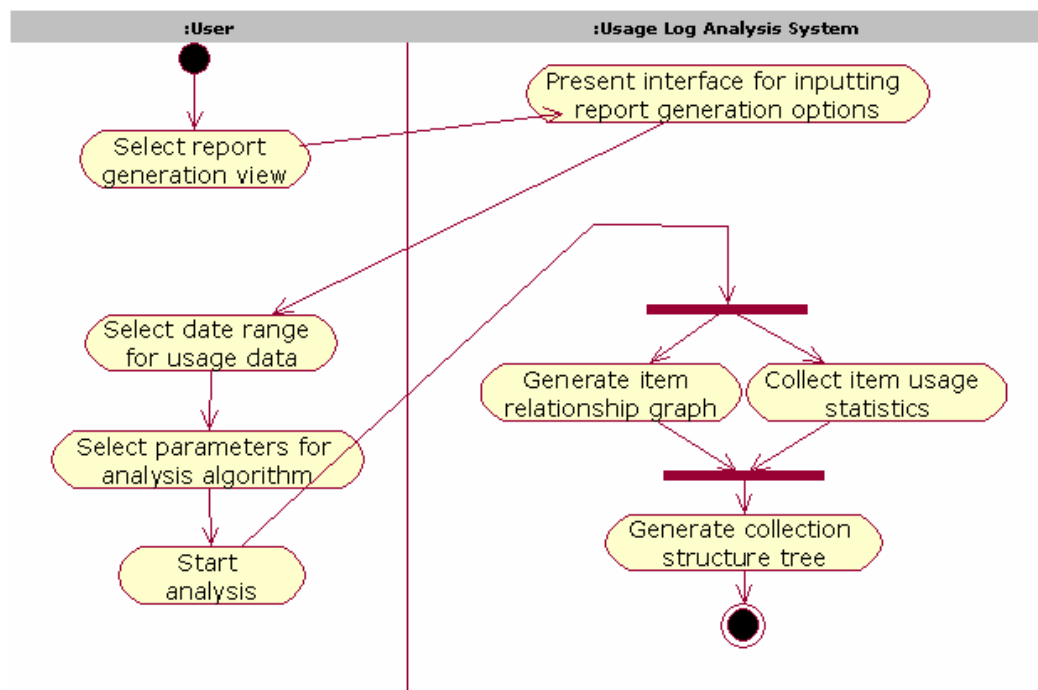


Figure 7 Activity diagram for Generate analysis report

### 2.3.1.4 Remove analysis report (UC-04)

<b>Identifier</b>	UC-04
<b>Use-Case Name</b>	Remove analysis report
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user can remove analysis report

<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-02 (OCD 4.3.2)
<b>Requirements</b>	IR-4 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	User selects id of the analysis report to be deleted and pressed remove button
<b>User Interface</b>	User is presented with a scrollable selectable list of previously generated analysis reports and a button to initiate removal
<b>Pre-conditions</b>	One or more analysis reports have been generated
<b>Post-conditions</b>	System database does not contain any analysis information that was associated with the removed report
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

Table 4 Use-Case Description for Remove analysis report

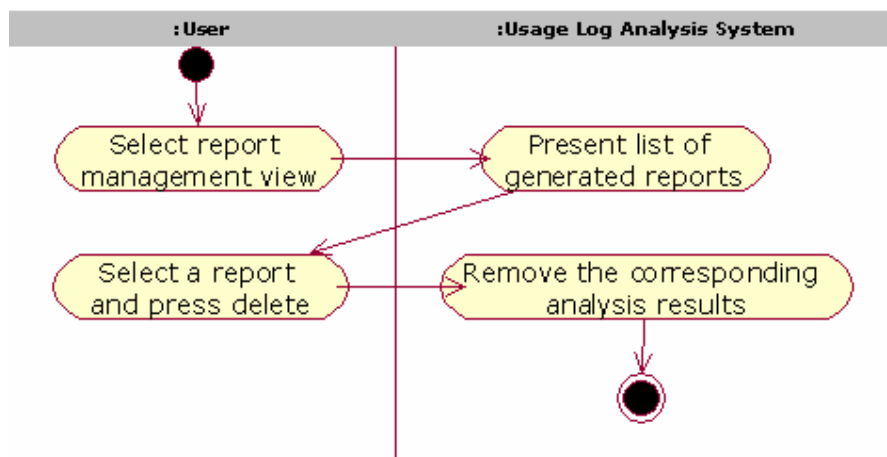


Figure 8 Activity diagram for Remove analysis report

### 2.3.1.5 Open analysis report (UC-05)

<b>Identifier</b>	UC-05
<b>Use-Case Name</b>	Open analysis report
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user selects an analysis report in order to visualize it
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-03 (OCD 4.3.2)
<b>Requirements</b>	SR-6 (SSRD 3.2.1) IR-4 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	In order to visualize an analysis report user selects it from a list of generated reports
<b>User Interface</b>	User is presented with a scrollable selectable list of previously generated analysis reports and a button to visualize them.
<b>Pre-conditions</b>	One or more analysis reports have been generated
<b>Post-conditions</b>	A new window with 3d visualization of the selected analysis report is opened
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

**Table 5 Use-Case Description for Open analysis report**

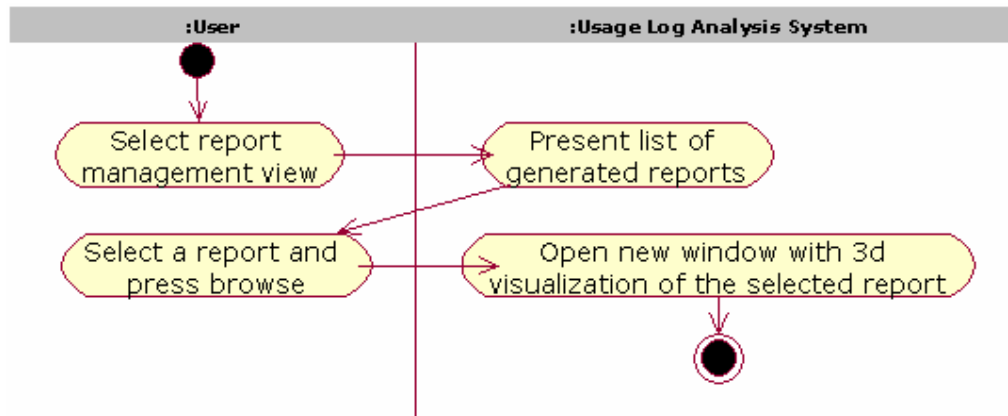


Figure 9 Activity diagram for Visualize analysis report

### 2.3.1.6 Browse analysis report (UC-05)

<b>Identifier</b>	UC-06
<b>Use-Case Name</b>	Browse analysis report
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user can interact with the visualization of the analysis report
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-03 (OCD 4.3.2)
<b>Requirements</b>	SR-6 (SSRD 3.2.1) IR-5 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	User can browse the 3d visualization of analysis report by panning and rotating the view using left and right mouse buttons correspondingly. User can examine usage statistic for a particular node by selecting it in the view.
<b>User Interface</b>	Collection structure tree is presented in a 3d hyperbolic interactive view.
<b>Pre-conditions</b>	Analysis report has been generated and selected for visualization.

<b>Post-conditions</b>	System changes the view and presented information based on the dynamic user input.
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

Table 6 Use-Case description for Browse analysis report

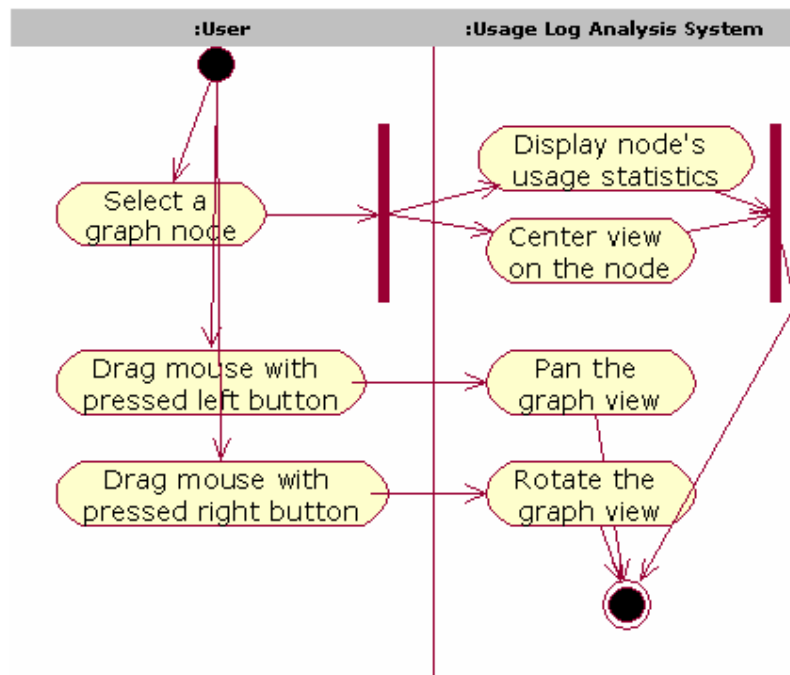


Figure 10 Activity diagram for Browse analysis report

## 2.3.2 Modes of Operation

### 2.3.2.1 Operational mode

The proposed system has only one mode of operation in which it will provide all the capabilities defined in OCD 4.3.

Capability	Processes	Mode Impact
Manage usage log data	<ul style="list-style-type: none"> <li>Import usage data</li> <li>Remove usage data</li> </ul>	N / A
Log Analysis	<ul style="list-style-type: none"> <li>Generate analysis report</li> <li>Remove analysis report</li> </ul>	N / A
Visualization of Log	<ul style="list-style-type: none"> <li>Open analysis report</li> </ul>	N / A

Analysis	<ul style="list-style-type: none"> <li>Browse analysis report</li> </ul>	
----------	--	--

**Table 7 Available Capabilities & Processes in Operational Mode**

## 2.4 L.O.S. Goals

<b>L.O.S requirement</b>	<b>Applies to</b>	<b>How</b>
LR-1: System Dependability - Stable data import/export	UC-01, UC-02	Equally
LR-2 : Usability – User-friendly interface for viewing item relationships and updating data	UC-05, UC-06	Equally
LR-3: Usability – Maximizing the usability of host resources	UC-3, UC-6	Equally
LR-4: Performance – Organizing data meaningfully for users	UC-1, UC-3	Equally
LR-5: Performance- data of current scale	UC-3	

**Table 8 L.O.S Goals**

## 2.5 Rules

The proposed system will not change any current organization rules identified in OCD 4.5.4.

## 3. Architecture Design & Analysis

The architecture design and analysis provide a high-level architecture for the system and also explain the relationship between the components. This section will describe what are the work units (both hardware and software *components*) of the system and what the components are expected to do.

### 3.1 Structure

#### 3.1.1 Topology

The proposed system is a hybrid of simple and composite system. The system is simple since it's a standalone application that has a primitive "Hardware classifiers model" and "Deployment model". However software classifiers of this system are split into three layers: Interface, Controller and Model. Detailed description of each layer is provided in the following sections.

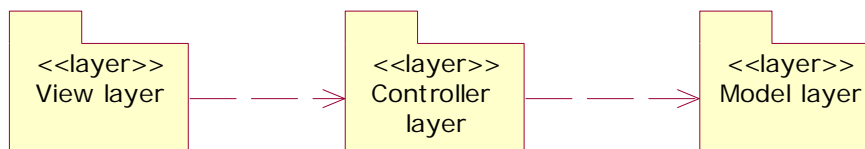


Figure 11 Topology of the system

##### 3.1.1.1 View layer

The interface layer contains components that allow system to communicate with the user. These components provide functionality for taking user input and displaying system's output. In particular interface layer contains component providing three dimensional hyperbolic visualization of a tree representing the Digital Archive collection.

Below is the list of interface components:

- Interface component
  - Importing usage data
  - Removing usage data
  - Generating new report
  - Removing report
  - Opening report

- Visualization component
  - Three dimensional view of collection tree
  - Display of usage statistics for selected item

### **3.1.1.2 Controller layer**

Controller layer receives requests from Interface layer, performs requested operations, sends updates to the Model layer and notifies Interface about completion of operation. This layer includes the following components:

- Dispatcher
- Data import
- Relationship & statistics generation
  - Generate item relationship graph
  - Collect item usage statistics
- Tree generation
  - Cluster item relationship graph
  - Perform centrality analysis
  - Generate item tree

### **3.1.1.3 Model layer**

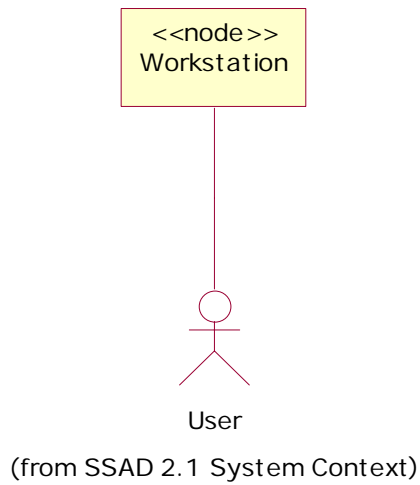
Model layer provides persistent representation of the system states. This layer contains components that provide storage for input data and intermediate operational results of components. This layer contains one Database component that provides four interfaces to other components as follows:

- Usage data table
- Relationships table
- Statistics table
- Tree table

## **3.1.2 Hardware Classifier Model**

This section describes hardware components that are either part of the system or on which this system will run and actors which will interact with the hardware components. The proposed system is a standalone application that will be installed

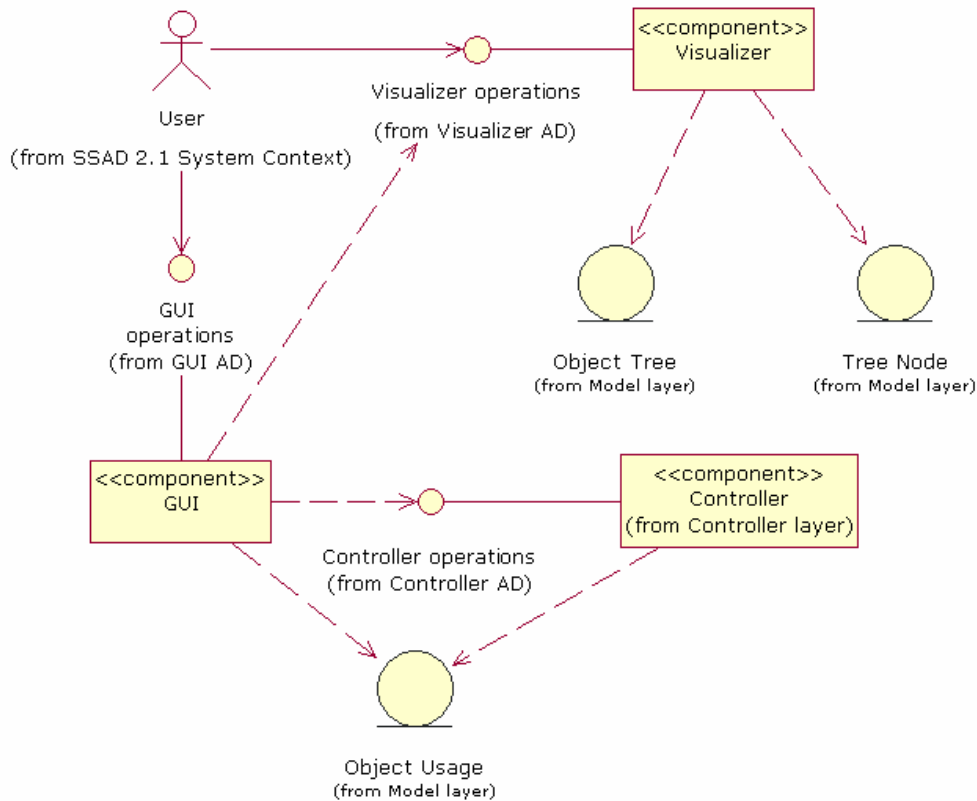
and operated on a local workstation. The system does not have remotely deployed components and does not communication with other systems.



**Figure 12 Hardware Classifier Model**

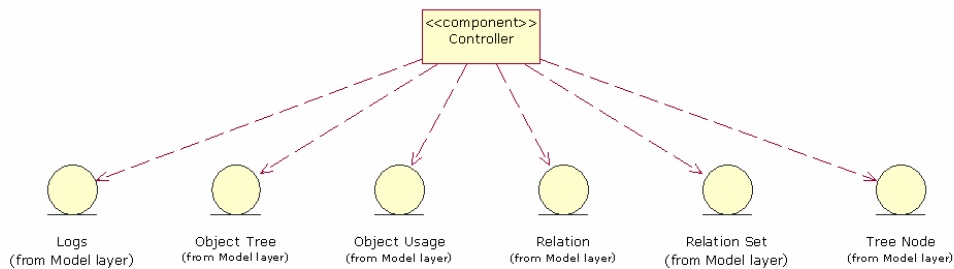
### 3.1.3 Software Classifier Model

This section describes in detail software component models for each layer of the system (Figure 12), interactions between them and with the user.



**Figure 13 View layer component diagram**

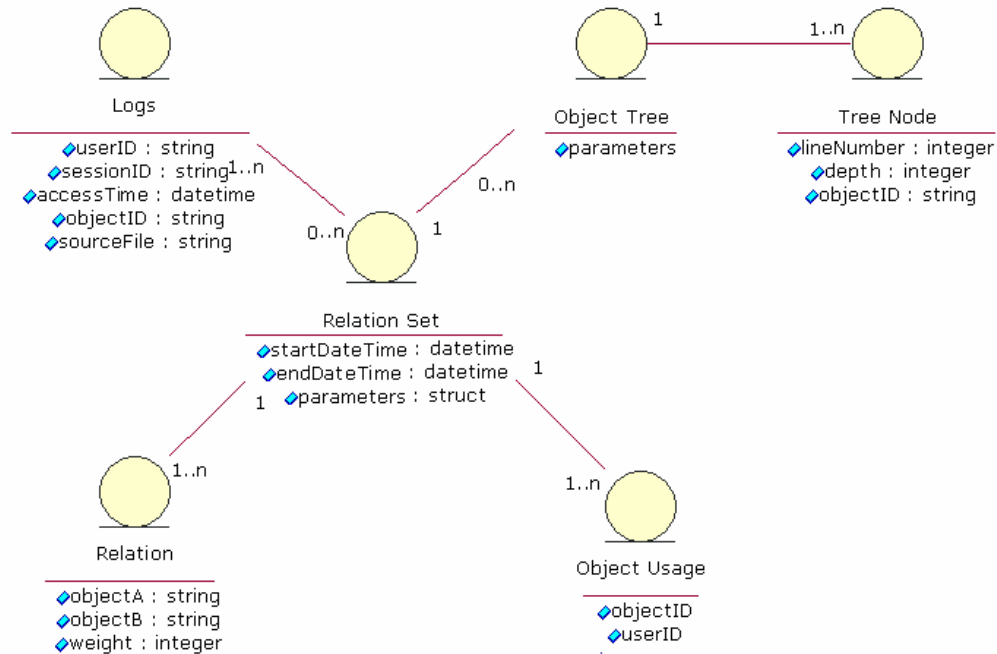
View layer contains components that provide graphical representation of system’s state and allow user to interact with the system. The Interface component of the View layer provides user with ability to pass input data and parameters to the system and invoke the system’s functionality. Visualizer component provides interactive three-dimensional display of generated item relations and usage statistics.



**Figure 14 Controller layer component diagram**

Controller layer contains components that provide data management and data mining capabilities. Retrievals manager component is responsible for parsing usage data from

input log file and storing into the Retrieval Table. Also this component provides capability to remove retrieval data previously imported and to get list of imported retrieval files. Reports manager provides capabilities for generation and removal of reports and also for retrieval of a list of generated reports.



**Figure 15 Model software component diagram**

The “Model” layer contains “Database” component that provides several interfaces for data storage, searching and retrieval in form of SQL tables. Table design specializes SQL interface by specifying number of columns, their names and their data types.

### 3.1.4 Deployment Model

Deployment model describes component and connector configuration(s) that make a working version of the system. In this configuration, it describes the instances of hardware and software component that participate in the configuration, the allocation of software components to the hardware components.

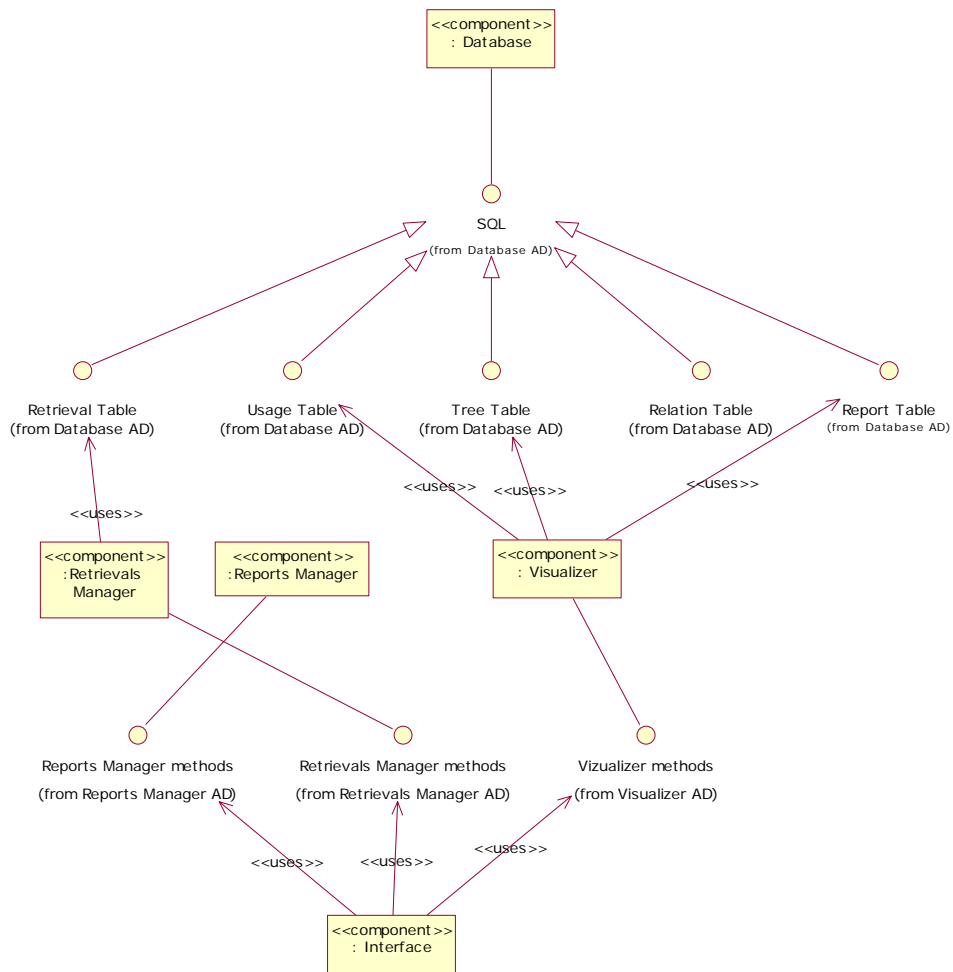


Figure 16 Deployment Model

### 3.1.5 Hardware Component Classifiers

#### 3.1.5.1 Classifier: Workstation (HCC-01)

This section describes the only hardware component that will be used by the system – which is Workstation.

##### 3.1.5.1.1 Purpose

The purpose of this hardware component is to serve as a host for the proposed application. This is the environment where all the software components of the system will be installed and operated.

### **3.1.5.1.2 L.O.S. Goals**

None of the LOS requirements apply here.

## **3.1.6 Hardware Connector Classifiers**

No Hardware Connector Classifiers have been defined because the proposed system is a standalone application.

## **3.1.7 Software Component Classifiers**

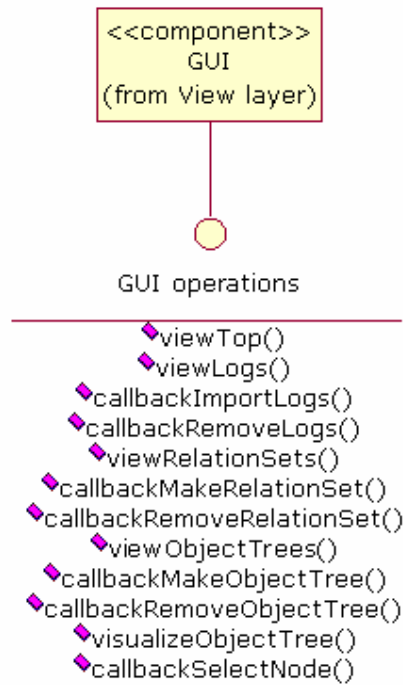
### **3.1.7.1 GUI (SCC-01)**

#### **3.1.7.1.1 Purpose**

The purpose of this component is to provide Graphical User Interface that allows user to input necessary information and invoke different system's functionalities.

#### **3.1.7.1.2 Interface**

The GUI component will have one interface that provides operations for generating different system views and for invoking system capabilities. The following operations are available:



**Figure 17 GUI Interface**

#### 3.1.7.1.2.1 viewTop

This function constructs and renders top view of the system menus and forms. This function does not take any parameters. This function should present interface to select one of the available sub-views: Logs view, Relation Sets view, Object Trees view. It should also open a default view (Logs view). Precondition: system is initialized properly. Postcondition: default view is open; interface to select available view is presented.

#### 3.1.7.1.2.2 viewLogs

Parameters: none

Preconditions: Top view of the system was drawn. System was just initialized or Logs view was selected by the user.

Postconditions: List of imported Logs is presented. Interface to import and remove Logs is presented.

#### 3.1.7.1.2.3 callbackImportLogs

Parameters: name of the source Log File

Preconditions: Logs view selected. Import Logs form is submitted

Postconditions: Controller::importLogs operation is called.

#### **3.1.7.1.2.4 callbackRemoveLogs**

Parameters: name of the source Log File

Preconditions: Logs view selected. Log File with the specified name was previously imported

Postconditions: Controller::removeLogs operations is called

#### **3.1.7.1.2.5 viewRelationSets**

Parameters: none

Preconditions: Top view of the system was drawn. User selected Relation Sets view.

Postconditions: New Relation Set form is drawn. List of previously generated Relations Sets is drawn. Interface to remove a Relation Set is presented.

#### **3.1.7.1.2.6 callbackMakeRelationSet**

Parameters: Filled New Relation Set form.

Preconditions: Relations Sets view was drawn. New Relation Set form was submitted.

Postconditions: Controller::makeRelationSet operation is called

#### **3.1.7.1.2.7 callbackRemoveRelationSet**

Parameters: Filled Remove Relation Set form.

Preconditions: Relation Set view was drawn. User selected to delete a Relation Set

Postconditions: Controlle::removeRelationSet operation is called.

#### **3.1.7.1.2.8 viewObjectTrees**

Parameters: none

Preconditions: Top view of the system was drawn. User selected Object Trees view.

Postconditions: New Object Tree form is presented. List of generated object trees is presented. Interface to open or delete an Object Tree is presented.

#### **3.1.7.1.2.9 callbackMakeObjectTree**

Parameters: Filled New Object Tree form

Preconditions: Object Trees view was drawn. User submitted New Object Tree form.

Postconditions: Controller::makeObjectTree operation is called

**3.1.7.1.2.10 callbackRemoveObjectTree**

Parameters: Filled Remove Object Tree form.

Preconditions: Object Trees view was drawn. User submitted Remove Object Tree form.

Postconditions: Controller::removeObjectTree operations is called

**3.1.7.1.2.11 callbackOpenObjectTree**

Parameters: Filled Open Object Tree form.

Preconditions: Object Trees view was drawn. User submitted Open Object Tree form.

Postcondition: H3Viewer::setGraph operation is called

**3.1.7.1.2.12 callbackSelectObject**

Parameters: Object name.

Preconditions: Object Tree was opened and visualized by H3Viewer. User clicked on a tree node.

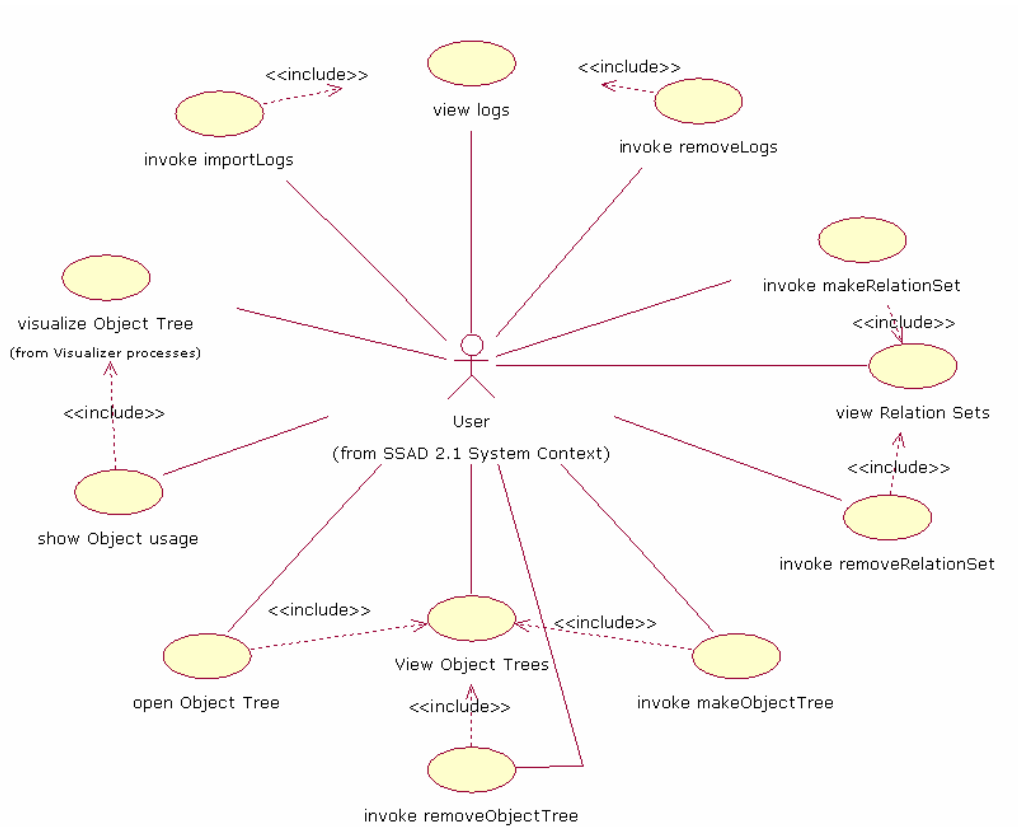
Postcondition: Object usage information is displayed.

**3.1.7.1.3 Parameters**

The GUI component is a static component which has a predefined behavior and therefore does not have any parameters.

**3.1.7.1.4 Behavior**

**3.1.7.1.4.1 Processes**



**Figure 18 GUI processes use-case diagram**

**3.1.7.1.4.1.1 view logs**

<b>Identifier</b>	PC-1.1
<b>Use-Case Name</b>	view logs
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user would select Logs view
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development</b>	RLCA

<b>Status</b>	
<b>Overview</b>	After the system is initialized the user should be presented with the top level view of the system and interface to select available sub-views. After user selects logs view system
<b>User Interface</b>	User is presented with a scrollable selectable list of imported logs with source file name displayed. There should be a field to specify path to a new Log file to import and import button. Also there should be a delete button to remove Logs of the file selected from the list.
<b>Pre-conditions</b>	Top view of the system is drawn
<b>Post-conditions</b>	Logs view is presented to the user as described.
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

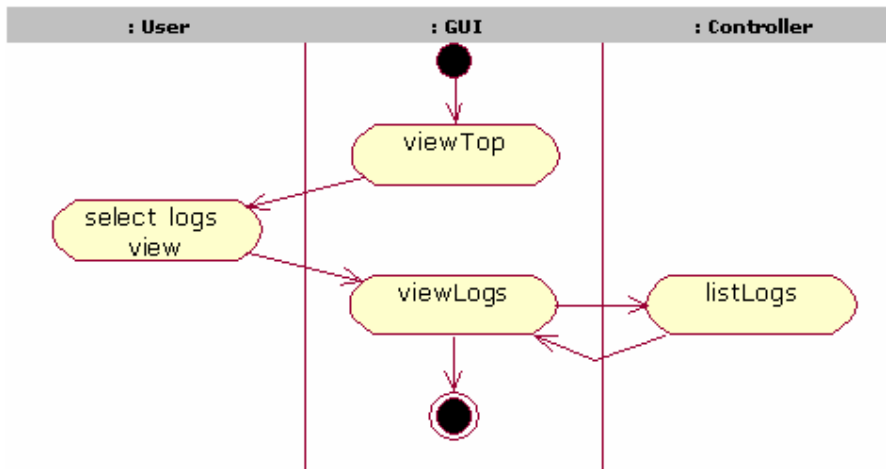


Figure 19 View logs process Activity diagram

#### 3.1.7.1.4.1.2 invoke importLogs

<b>Identifier</b>	PC-1.2
<b>Use-Case Name</b>	invoke importLogs
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user invokes importing of a new Log File
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)

<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	User should be able to specify location of a Log File and import Logs that are contained in it
<b>User Interface</b>	Text field to input location path of the file. Browse button to open a standard file browsing/selecting interface. Import button to invoke importing process.
<b>Pre-conditions</b>	Logs view was drawn. Source Log File is available from the on the local machine.
<b>Post-conditions</b>	Operation importLogs of the Controller component is invoked
<b>Specializes</b>	None
<b>Includes</b>	View logs
<b>Extends</b>	None
<b>Extension Points</b>	None

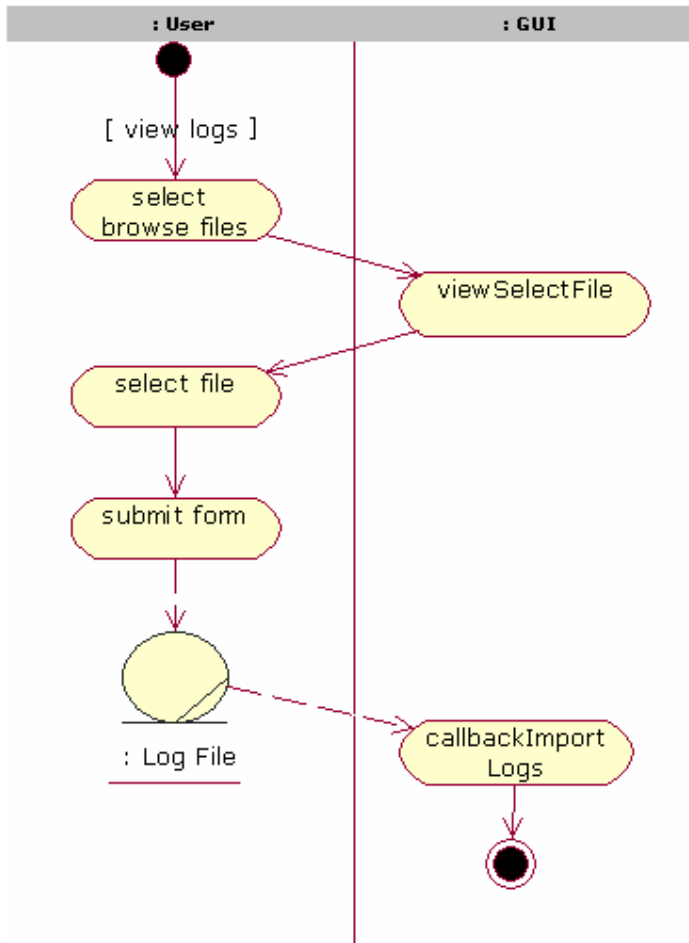


Figure 20 Invoke importLogs Activity Diagram

3.1.7.1.4.1.3 Invoke removeLogs

<b>Identifier</b>	PC-1.3
<b>Use-Case Name</b>	Invoke removeLogs
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user invokes deletion of Logs that were previously imported from specific Log File.
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No

<b>Development Status</b>	RLCA
<b>Overview</b>	Being at the Logs view user selects a Log File name from the list of previously imported Logs and presses deleted, which initiates removing of Log records.
<b>User Interface</b>	Selectable list of imported Log Files and a delete button.
<b>Pre-conditions</b>	Logs view was drawn. At least one Log File has been imported.
<b>Post-conditions</b>	Operation removeLogs of the Controller component is called.
<b>Specializes</b>	None
<b>Includes</b>	View logs
<b>Extends</b>	None
<b>Extension Points</b>	None

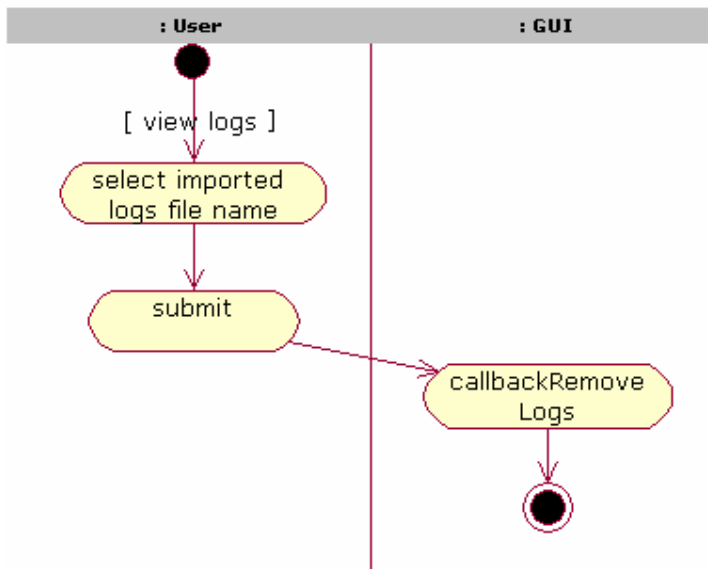


Figure 21 Invoke removeLogs Activity Diagram

3.1.7.1.4.1.4 View Relation Sets

<b>Identifier</b>	PC-1.4
<b>Use-Case Name</b>	View Relation Sets
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how Relation Sets view is selected by the user.
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)

<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	Being at the top view user should be able to select the Relation Sets view. After the view is selected system should present interface for observing which relation sets have been created, interface to create new Relation Sets and remove old ones.
<b>User Interface</b>	Selectable, scrollable list of created Relation Sets. For each set the associated information should be presented: start, end dates, analysis parameters and generation notes.
<b>Pre-conditions</b>	Top view of the system was drawn. User selected Relation Sets view.
<b>Post-conditions</b>	Described interface is presented.
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

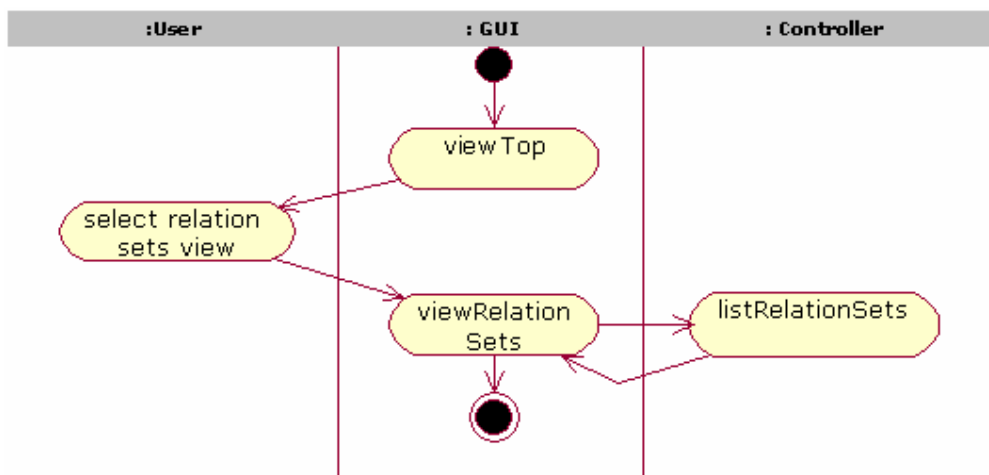


Figure 22 View Relation Sets Activity Diagram

## 3.1.7.1.4.1.5 Invoke makeRelationSet

<b>Identifier</b>	PC-1.5
-------------------	--------

<b>Use-Case Name</b>	Invoke makeRelationSet
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user invokes creation of a new Relation Set
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	Being at the Relation Sets view user should be able to enter parameters for creating new Relation Set: date range for source Log data, analysis parameters. User then submits the filled form to initiate Relation Set creation.
<b>User Interface</b>	Modifiable text fields. Date selection menu using dates available from currently imported Log Data. Create button.
<b>Pre-conditions</b>	Relation Set view was drawn. User submitted New Relation Set form.
<b>Post-conditions</b>	Operation makeRelationSet of Controller component is called.
<b>Specializes</b>	None
<b>Includes</b>	View Relation Sets
<b>Extends</b>	None
<b>Extension Points</b>	None

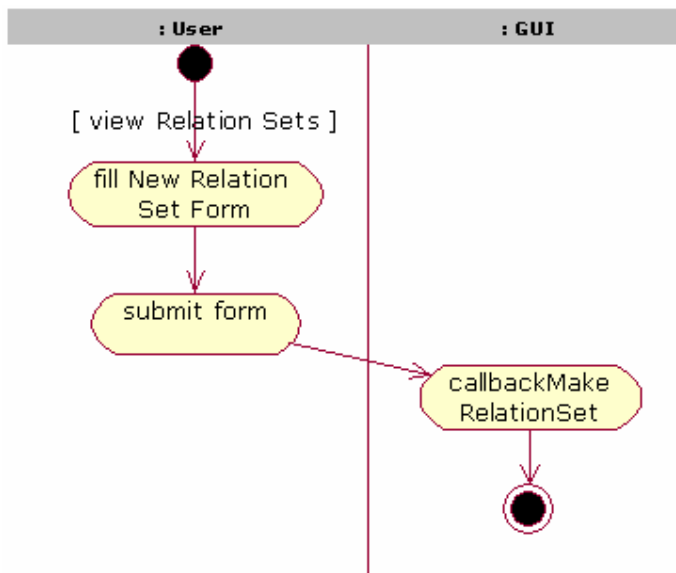


Figure 23 Invoke makeRelationSet Activity Diagram

## 3.1.7.1.4.1.6 Invoke removeRelationSet

<b>Identifier</b>	PC-1.6
<b>Use-Case Name</b>	Invoke makeRelationSet
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user invokes removal of Relation Set
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	Being at the Relation Sets view user selects a Relation Set from the list of previously generated sets. User presses remove button to initiate removal process.
<b>User Interface</b>	Selectable scrollable list of previously generated Relation Sets. Remove button to generated the removal
<b>Pre-conditions</b>	Relation Set view was drawn. At least one Relation Set has been previously generated. User pressed remove button.

<b>Post-conditions</b>	Operation removeRelationSet of Controller component is called.
<b>Specializes</b>	None
<b>Includes</b>	View Relation Sets
<b>Extends</b>	None
<b>Extension Points</b>	None

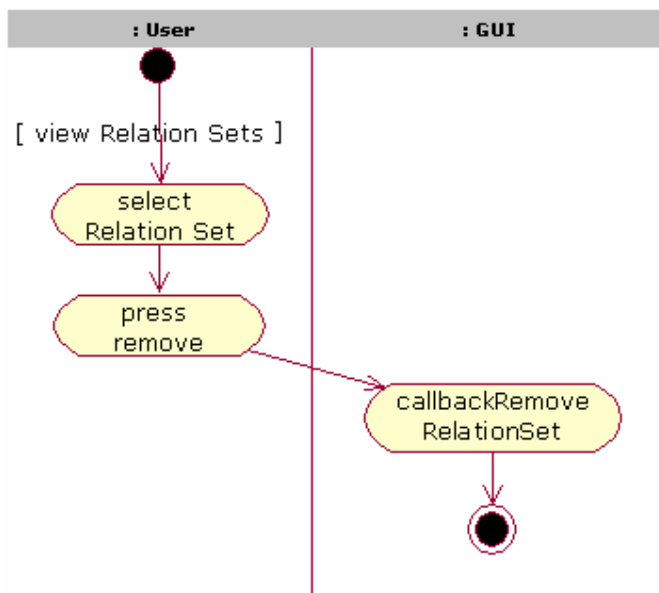


Figure 24 Invoke removeRelationSet Activity Diagram

3.1.7.1.4.1.7 View Object Trees

<b>Identifier</b>	PC-1.7
<b>Use-Case Name</b>	View Object Trees
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user selects Object Trees sub-view of the system
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No

<b>Development Status</b>	RLCA
<b>Overview</b>	Being at the top view of the system user selects the Object Trees view and then is presented with the interface to create, remove and visualize object trees.
<b>User Interface</b>	User is presented with a selectable scrollable list of previously generated object trees. For each object tree the associated information is displayed: source Relation Set and analysis parameters. Remove button is provided to initiate removal of an Object Tree. New Object Tree Form is provided. Visualize button is provided to open Object Tree for interactive visualization
<b>Pre-conditions</b>	Top view of the system was drawn. User selected Object Trees sub-view.
<b>Post-conditions</b>	Interface is presented as described above.
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

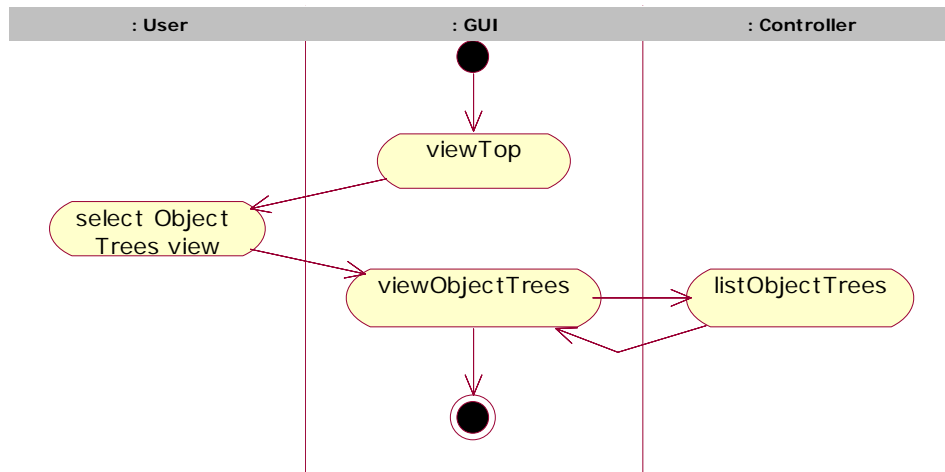


Figure 25 View Object Trees Activity Diagram

3.1.7.1.4.1.8 Invoke makeObjectTree

<b>Identifier</b>	PC-1.8
<b>Use-Case Name</b>	Invoke makeObjectTree

<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user invokes creation of a new Object Tree
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	Being at the Object Trees view user fills the New Object Tree form and submits it to initiate generation.
<b>User Interface</b>	Selection menu of available Relation Sets. Modifiable form fields for analysis parameters. Create button to initiate creation.
<b>Pre-conditions</b>	View Object Trees was drawn. User submitted the New Object Tree form.
<b>Post-conditions</b>	Operation makeObjectTree of the Controller component is called
<b>Specializes</b>	None
<b>Includes</b>	View Object Trees
<b>Extends</b>	None
<b>Extension Points</b>	None

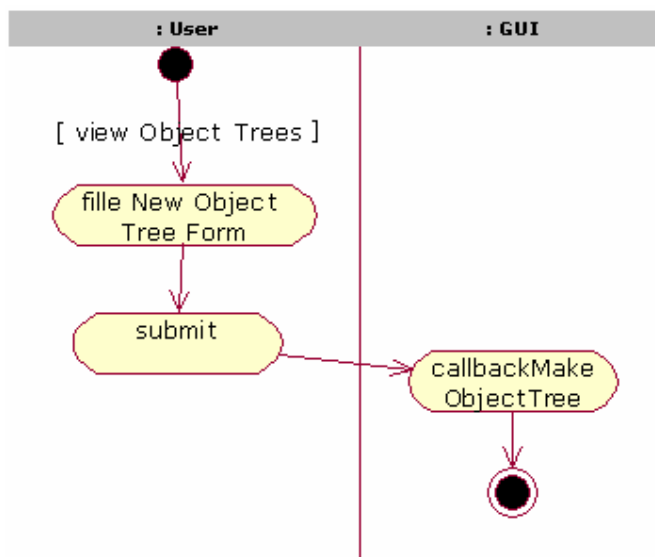


Figure 26 Invoke makeObjectTree Activity Diagram

## 3.1.7.1.4.1.9 Invoke removeObjectTree

<b>Identifier</b>	PC-1.9
<b>Use-Case Name</b>	Invoke removeObjectTree
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user invokes removal of previously generated object tree.
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	Being at the Object Trees view user select a previously generated Object Tree from a list and presses remove to initiate removal.
<b>User Interface</b>	Selectable scrollable list of previously generated Object Trees. Remove button
<b>Pre-conditions</b>	View Object Trees was drawn. User pressed remove button.
<b>Post-conditions</b>	Operation removeObjectTree of the Controller component is

	called
<b>Specializes</b>	None
<b>Includes</b>	View Object Trees
<b>Extends</b>	None
<b>Extension Points</b>	None

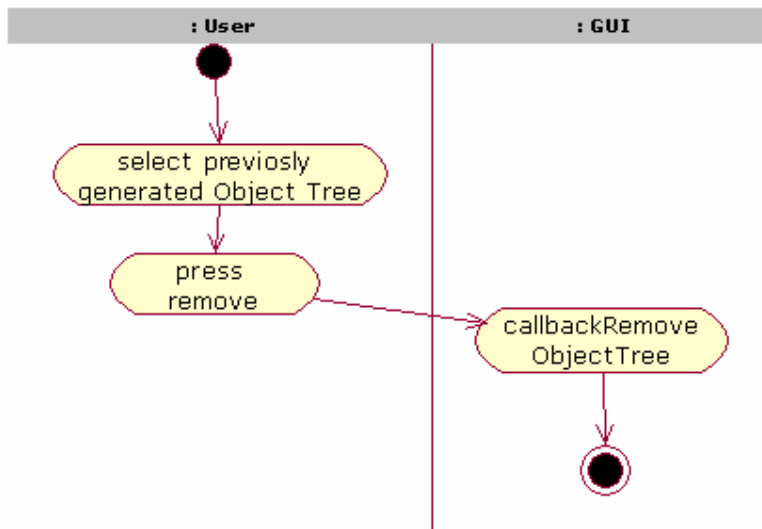


Figure 27 Invoke removeObjectTree

3.1.7.1.4.1.10 Open Object Tree

<b>Identifier</b>	PC-1.10
<b>Use-Case Name</b>	Open Object Tree
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how user opens previously generated Object Tree for visualization
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA

<b>Overview</b>	Being at the Object Trees view user select a previously generated Object Tree from a list and presses open button to initiate visualization. System should create a separate window / tab where the visualization will be done and which could be closed later. Within that window h3viewer component should be launched. A side window is setup to display usage information for a selected object.
<b>User Interface</b>	Selectable scrollable list of previously generated Object Trees. Open button. Visualization is opened in new window or tab. Side window is setup for Object Usage information.
<b>Pre-conditions</b>	View Object Trees was drawn. User pressed open button.
<b>Post-conditions</b>	New window tab is initialized. H3Viewer component is set up within that window. Operation setGraph of the H3Viewer component is called.
<b>Specializes</b>	None
<b>Includes</b>	View Object Trees
<b>Extends</b>	None
<b>Extension Points</b>	None

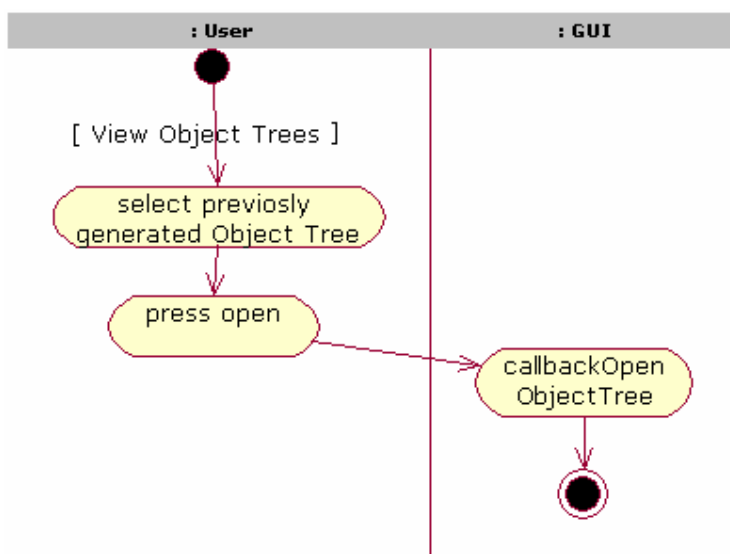


Figure 28 Open Object Tree Activity Diagram

#### 3.1.7.1.4.2 Modes of Operation

This component as the whole system has only one operational mode.

### 3.1.7.1.5 L.O.S. Goals

<b>L.O.S. Requirement:</b>	LR-2 : Usability – User-friendly interface for viewing item relationships and updating data
<b>Description:</b>	This component should provide user-friendly interfaces for viewing and updating data.
<b>Measurable:</b>	User will provide usability feedback
<b>Relevant:</b>	SR-2: Remove usage data imported from the specified log file. SR-3: Relationship generation SR-4: Generate collection structure tree. SR-6: Visualization. OCD 4.4 LS-2: Usability
<b>Specific:</b>	Specifies which requirements and goals are realized.

Figure 29 LOS Goals for GUI Component Classifier

### 3.1.7.1.6 Constraints

There are no additional constraints.

## 3.1.7.2 Visualizer (SCC-02)

### 3.1.7.2.1 Purpose

This component provides 3d hyperbolic visualization of object trees.

### 3.1.7.2.2 Interface

This component has one interface that provides the following operations.

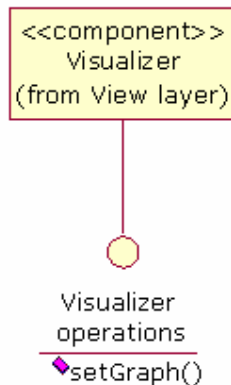


Figure 30 Visualizer interface

**3.1.7.2.2.1 setGraph**

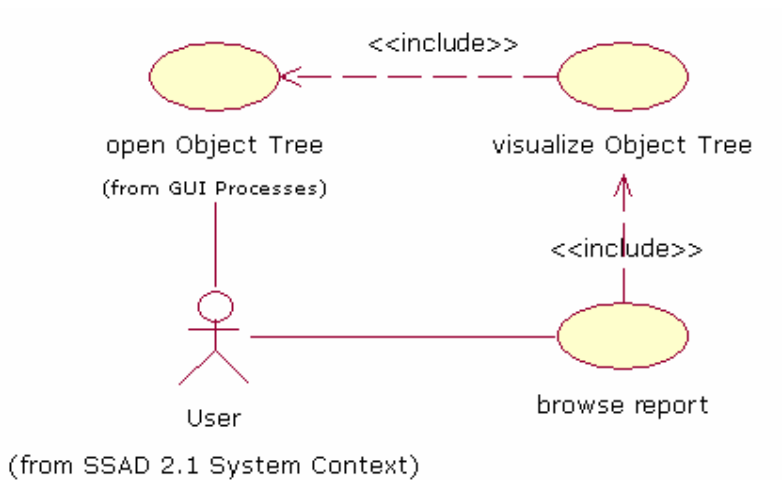
Parameters: Object Tree id.

Preconditions: Visualizer was properly initialized. At least one Object Tree was generated.

Postconditions: 3d hyperbolic view of the specified Object Tree is presented in the Visualizer’s frame.

**3.1.7.2.3 Behavior**

**3.1.7.2.3.1 Processes**



**Figure 31 Visualizer processes Use-Case diagram**

**3.1.7.2.3.1.1 Visualize Object Tree**

<b>Identifier</b>	PC-2.1
<b>Use-Case Name</b>	Visualize Object Tree
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how system would visualize an Object Tree
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No

<b>Development Status</b>	RLCA
<b>Overview</b>	After user selects an Object Tree from a list of generated tree and selects open system initializes a Visualization component in a separate frame and passes it a reference to Object Tree.
<b>User Interface</b>	The visualization component draws and a 3d hyperbolic view of an object tree and provides interface for interactive browsing.
<b>Pre-conditions</b>	Visualizer component was properly initialized. Object Tree has been generated and reference to it is available.
<b>Post-conditions</b>	3d hyperbolic graph view is presented to the user.
<b>Specializes</b>	None
<b>Includes</b>	Open Object Tree
<b>Extends</b>	None
<b>Extension Points</b>	None

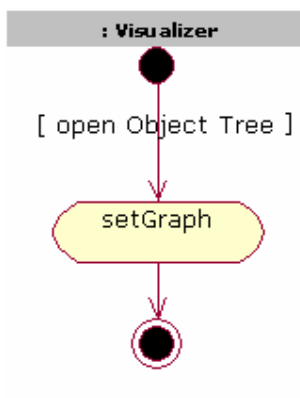


Figure 32 Visualize Object Tree Activity Diagram

#### 3.1.7.2.3.1.2 Browse Object Tree

<b>Identifier</b>	PC-2.2
<b>Use-Case Name</b>	Browse Object Tree
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates interactive interface provided by the visualizer component.
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)

<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	User changes the point of view for the graph by using mouse commands.
<b>User Interface</b>	3d hyperbolic view of the graph that responds to mouse commands by changing the Object Tree view.
<b>Pre-conditions</b>	Object Tree was opened in a Visualizer component
<b>Post-conditions</b>	The described interactive visualization interface is available.
<b>Specializes</b>	None
<b>Includes</b>	Visualize Object Tree
<b>Extends</b>	None
<b>Extension Points</b>	None

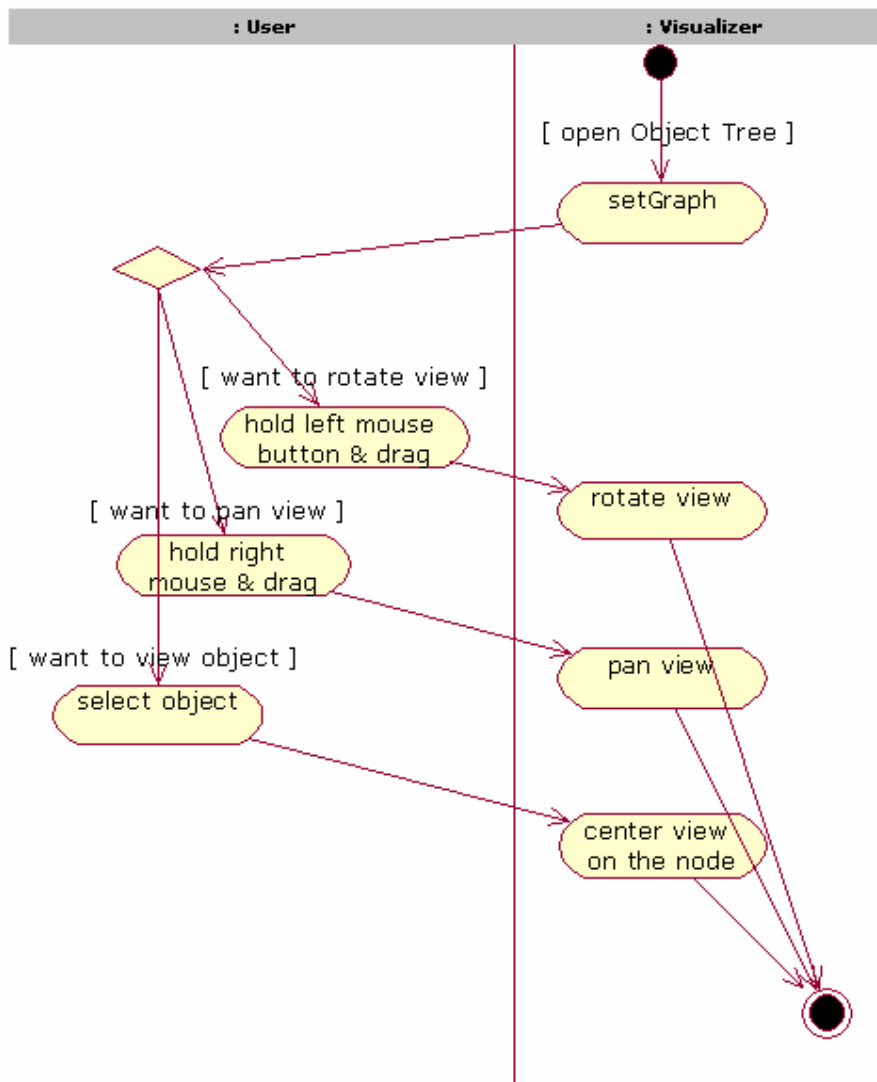


Figure 33 Browse Object Tree Activity Diagram

**3.1.7.2.3.2 Modes of Operation**

This component as the whole system has only one operational mode.

**3.1.7.2.4 L.O.S. Goals**

<p><b>L.O.S. Requirement:</b></p>	<p>LR-2 : Usability – User-friendly interface for viewing item relationships and updating data</p>
<p><b>Description:</b></p>	<p>The 3d hyperbolic interactive visualization of the Object Trees should be intuitive and informative enough so that user that had short training in system usage could use the system with 80%</p>

	efficiency.
<b>Measurable:</b>	Evaluation by instructed user and estimation efficiency of the user's work.
<b>Relevant:</b>	SR-6: Visualization. OCD 4.4 LS-2: Usability Ensures system usability and therefore satisfaction of the client.
<b>Specific:</b>	Specifies what user efficiency would be sufficient.

**Figure 34 LOS Goals for Visualizer component**

### 3.1.7.2.5 Constraints

There are no additional constrains.

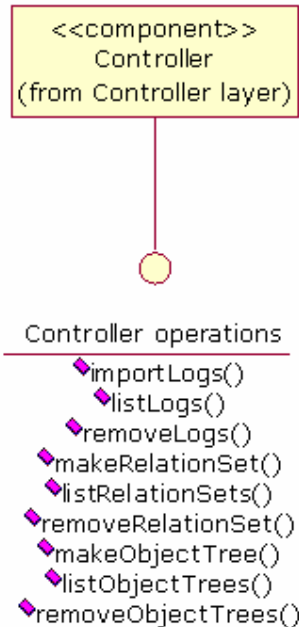
### 3.1.7.3 Controller (SCC-03)

#### 3.1.7.3.1 Purpose

This component provides functionality for processing input Log data and generating analysis data acceptable by visualization component. Also this component manages storage and retrieval of input data, temporary data and final analysis data.

#### 3.1.7.3.2 Interface

This component has one interface that provides the following operations.



**Figure 35 Controller interface**

#### 3.1.7.3.2.1 importLogs

Parameters: Path to the Log File to be imported.

Preconditions: Input Log File is available on local machine.

Postconditions: retrieval records from the Log File are parsed and stored by the system into the Logs database.

#### 3.1.7.3.2.2 listLogs

Parameters: None

Preconditions: System was initialized properly

Postconditions: List of Logs that have been imported previously is returned. If no Logs have been imported previously empty list is returned.

#### 3.1.7.3.2.3 removeLogs

Parameters: Name of the Log File that has been previously imported

Preconditions: At least one Log File has been previously imported.

Postconditions: All retrieval records that have been imported from the specified file are removed from the Logs database.

**3.1.7.3.2.4 makeRelationSet**

Parameters: start datetime, end datetime, algorithm parameters.

Preconditions: At least one Log File has been imported – so that input

Postconditions: set of relations is created according to the parameters specified and store to the Relation Set and Relations databases.

**3.1.7.3.2.5 listRelationSets**

Parameters: None

Preconditions: System was properly initialized.

Postconditions: list of previously generated relation sets is returned.

**3.1.7.3.2.6 removeRelationSet**

Parameters: RelationSet ID

Preconditions: At least one RelationSet has been generated previously

Postconditions: All records associated with the specified RelationSet are removed from the Relation Set, and Relations databases.

**3.1.7.3.2.7 makeObjectTree**

Parameters: RelationSet ID, algorithm parameters

Preconditions: At least one Relation Set has been generated.

Postconditions: Obejct tree is generated according to the algorithm and stored into Obejct Tree and Tree Node databases.

**3.1.7.3.2.8 listObjectTrees**

Parameters: None

Preconditions: System was properly initialized

Postconditions: List of previously generated Object Trees is returned

**3.1.7.3.2.9 removeObjectTree**

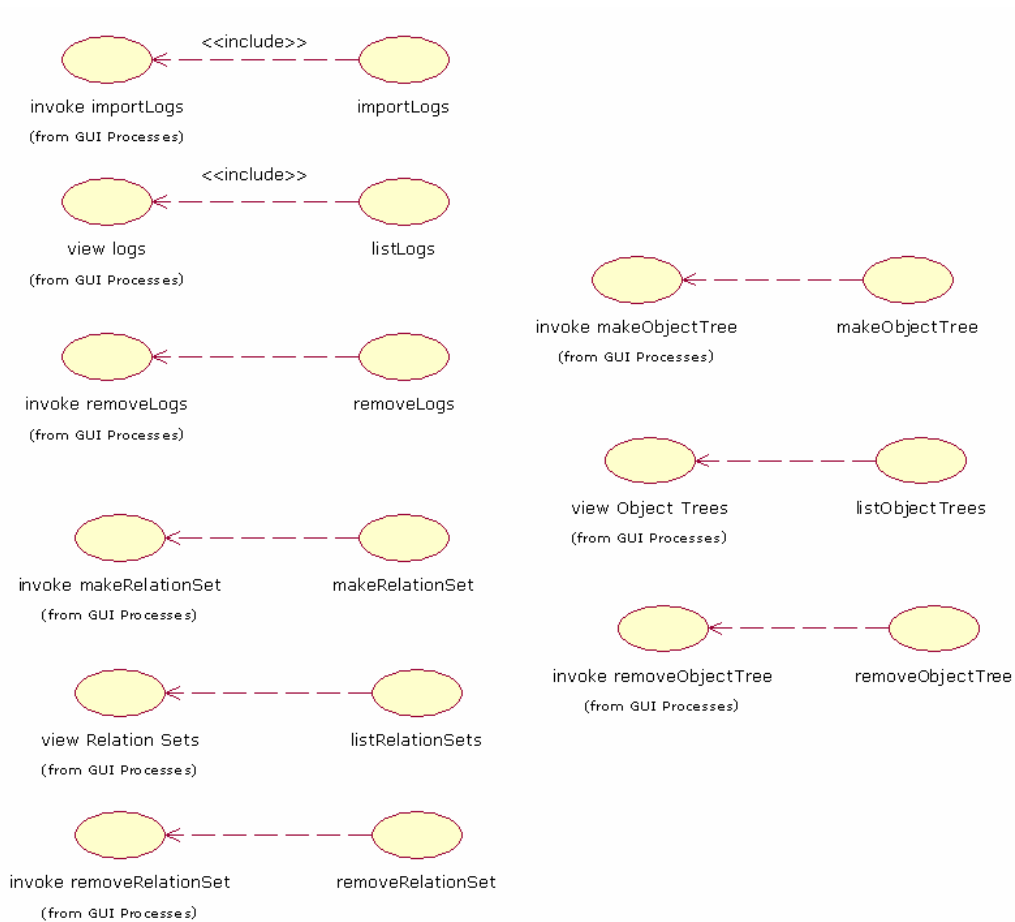
Parameters: Object Tree ID

Preconditions: At least one Object Tree has been generated

Postconditions: All records associated with the specified Object Tree are removed from Object Tree and Tree Node databases.

**3.1.7.3.3 Behavior**

**3.1.7.3.3.1 Processes**



**Figure 36 Controller Processes Use-Case Diagram**

**3.1.7.3.3.1.1 importLogs**

<b>Identifier</b>	PC-3.1
<b>Use-Case Name</b>	Import Logs
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how the system would import input Log Data
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally</b>	No

<b>Significant?</b>	
<b>Development Status</b>	RLCA
<b>Overview</b>	GUI component has received a request from the user to import certain Log File. GUI invokes importLog operation of the Controller component which parses the supplied data and stores it to the Logs database.
<b>User Interface</b>	None
<b>Pre-conditions</b>	System is initialized properly. Input Log File is available on the local machine
<b>Post-conditions</b>	Retrieval records contained in the supplied Log File are stored in the Logs database.
<b>Specializes</b>	None
<b>Includes</b>	Invoke Import Logs
<b>Extends</b>	None
<b>Extension Points</b>	None

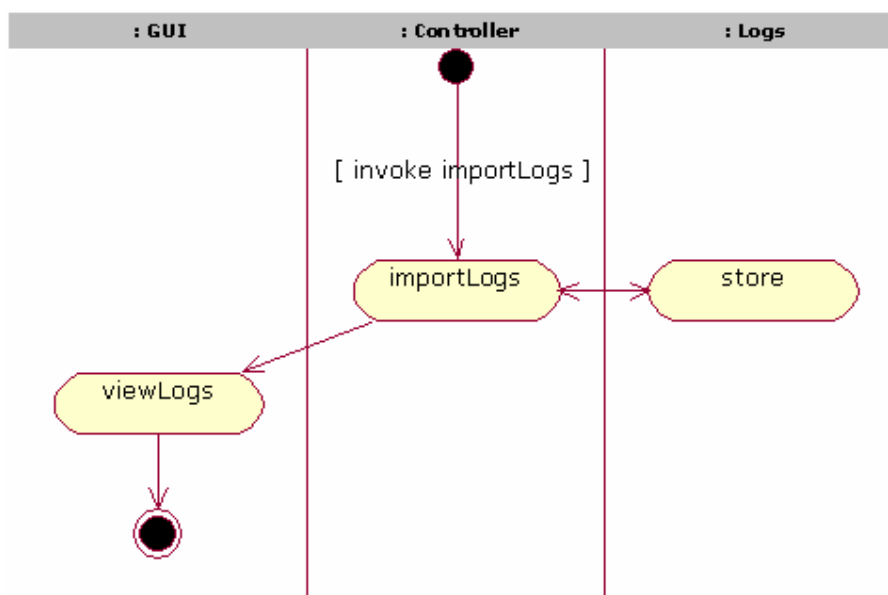


Figure 37 import Logs Activity Diagram

## 3.1.7.3.3.1.2 list Logs

<b>Identifier</b>	PC-3.2
<b>Use-Case Name</b>	List Logs

<b>Abstract</b>	No
<b>Purpose</b>	Demonstrated how system retrieves a list of previously imported logs
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	When User selects the view Logs view the system calls listLogs operation of the Controller component, which queries Logs database and returns list of imported logs.
<b>User Interface</b>	None
<b>Pre-conditions</b>	System is initialized properly
<b>Post-conditions</b>	List of imported Logs is returned
<b>Specializes</b>	None
<b>Includes</b>	View Logs
<b>Extends</b>	None
<b>Extension Points</b>	None

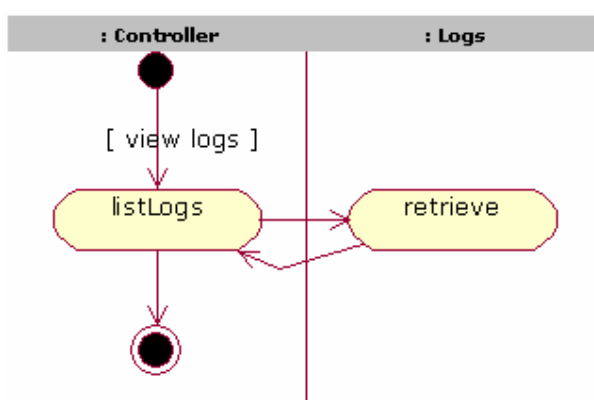


Figure 38 List Logs process Activity Diagram

**3.1.7.3.3.1.3 remove Logs**

<b>Identifier</b>	PC-3.3
<b>Use-Case Name</b>	Remove Logs
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrated how system removes previously imported logs
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	When removeLogs operation of the Controller component is called system accesses Logs database and removes all records that are associate with the supplied Log File name.
<b>User Interface</b>	None
<b>Pre-conditions</b>	System is initialized properly. Specified Log File was previously imported.
<b>Post-conditions</b>	Retrieval records associated with the specified Log File name are removed from the Logs database.
<b>Specializes</b>	None
<b>Includes</b>	Invoke removeLogs
<b>Extends</b>	None
<b>Extension Points</b>	None

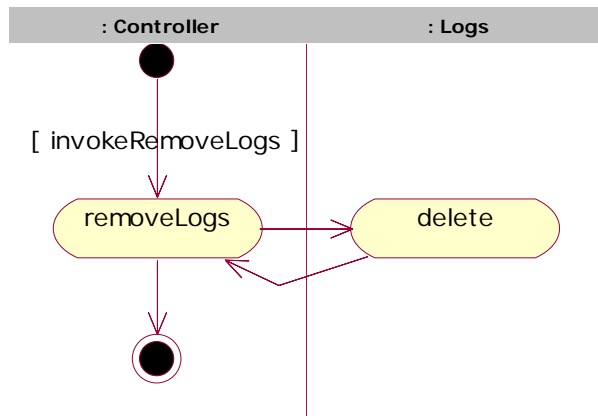


Figure 39 Remove Logs process Activity Diagram

3.1.7.3.3.1.4 makeRelationSet

<b>Identifier</b>	PC-3.4
<b>Use-Case Name</b>	Make Relation Set
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrated how system creates a Relation Set.
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	Supplied start date, end date, and algorithm options system generates a set of relations based on the Logs within the specified date range.
<b>User Interface</b>	None
<b>Pre-conditions</b>	At least one Log File has been imported. Parameters have been passed: start datetime, end datetime, algorithm options.
<b>Post-conditions</b>	Parameters that were passed for generation are stored as a record

	in Relation Set database. A set of relations is stores in the Relation database and is associated with the corresponding Relation Set.
<b>Specializes</b>	None
<b>Includes</b>	Invoke make RelationSet
<b>Extends</b>	None
<b>Extension Points</b>	None

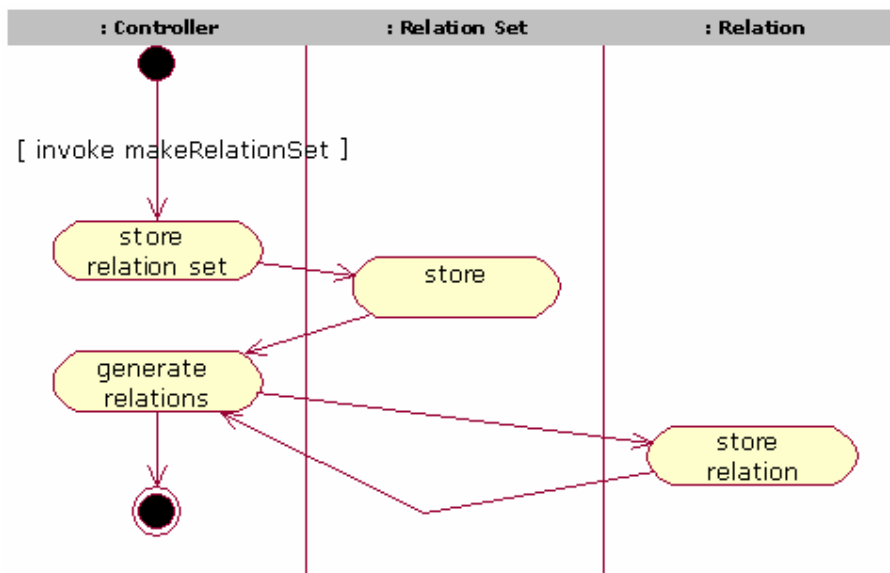


Figure 40 Make Relation Set Activity Diagram

3.1.7.3.3.1.5 List Relation Sets

<b>Identifier</b>	PC-3.5
<b>Use-Case Name</b>	List Relation Sets
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrated how system retrieves a list of previously generated Relation Sets
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No

<b>Development Status</b>	RLCA
<b>Overview</b>	When listRelationSets operation is called system accesses Relation Set database and retrieves a list of sets that have been generated so far.
<b>User Interface</b>	None
<b>Pre-conditions</b>	System is initialized properly
<b>Post-conditions</b>	List of previously generated Relation Sets is returned.
<b>Specializes</b>	None
<b>Includes</b>	View Relation Sets
<b>Extends</b>	None
<b>Extension Points</b>	None

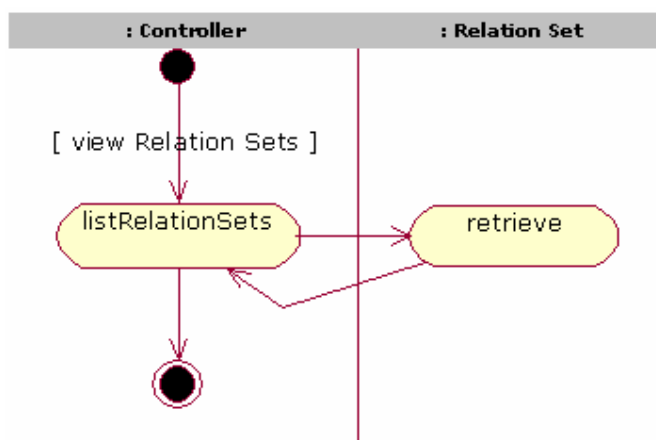


Figure 41 List Relation Sets Activity Diagram

### 3.1.7.3.3.1.6 Remove Relation Set

<b>Identifier</b>	PC-3.6
<b>Use-Case Name</b>	Remove Relation Set
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how system removes previously generated Relation Set
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None

<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	When removeRelationSet function is called the system accesses RelationSet and Relation databases and removes all records associated with the specified Relation Set.
<b>User Interface</b>	None
<b>Pre-conditions</b>	The specified Relation Set has been generated previously.
<b>Post-conditions</b>	All records associated with the specified Relation Set are deleted from the Relation Set and Relations databases.
<b>Specializes</b>	None
<b>Includes</b>	Invoke Remove Relations Set
<b>Extends</b>	None
<b>Extension Points</b>	None

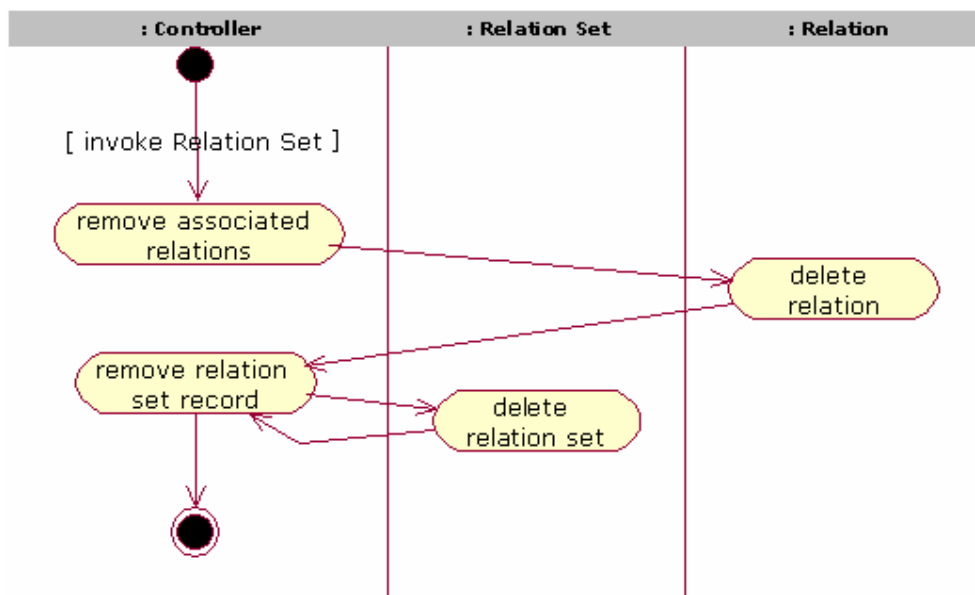


Figure 42 Remove Relation Set Activity Diagram

3.1.7.3.3.1.7 Make Object Tree

<b>Identifier</b>	PC-3.7
<b>Use-Case Name</b>	Make Object Tree

<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how system generates an Object Tree
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	When makeObjectTree operation is called and supplied the following arguments: Relation Set ID, algorithm options – the system generates a tree representation of the specified Relation Set according to the algorithm. This representation is save to Object Tree and Tree Node databases.
<b>User Interface</b>	None
<b>Pre-conditions</b>	Relation Set with the specified ID has been generated.
<b>Post-conditions</b>	Representation of Object Tree contained in Object Tree and Tree Node tables.
<b>Specializes</b>	None
<b>Includes</b>	Invoke Make Object Tree
<b>Extends</b>	None
<b>Extension Points</b>	None

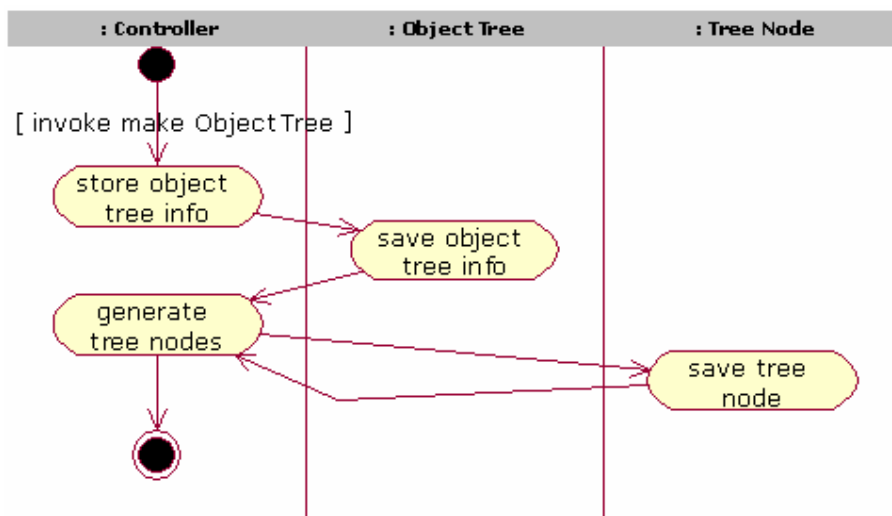


Figure 43 Make Object Tree Activity Diagram

3.1.7.3.3.1.8 List Object Trees

<b>Identifier</b>	PC-3.8
<b>Use-Case Name</b>	List Object Trees
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how system retrieves list of previously generated Object Trees.
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	When user selects Object Tree view system accesses Object Tree database and returns a list of previously generated Object Trees.
<b>User Interface</b>	None
<b>Pre-conditions</b>	System is initialized properly
<b>Post-conditions</b>	List of generated Object Trees is returned.
<b>Specializes</b>	None

<b>Includes</b>	View Object Trees
<b>Extends</b>	None
<b>Extension</b>	None
<b>Points</b>	

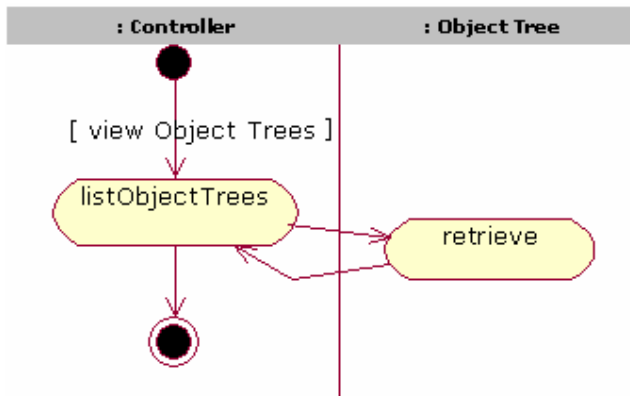


Figure 44 List Object Trees process Activity Diagram

3.1.7.3.3.1.9 Remove Object Tree

<b>Identifier</b>	PC-3.9
<b>Use-Case Name</b>	Remove Object Tree
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how system removes previously generated Object Tree
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3.1)
<b>Requirements</b>	SR-1 (SSRD 3.2.1), IR-1 (SSRD 4.1.1)
<b>Risks</b>	None
<b>High-Risk?</b>	No
<b>Architecturally Significant?</b>	No
<b>Development Status</b>	RLCA
<b>Overview</b>	When removeObjectTree function is called with Object Tree ID as a parameter the system accesses Object Tree and Tree Node databases and removes all records associated with the specified Object Tree ID
<b>User Interface</b>	None

<b>Pre-conditions</b>	The specified Object Tree ID has been previously generated
<b>Post-conditions</b>	All records associated with the specified Object Tree are removed from system databases.
<b>Specializes</b>	None
<b>Includes</b>	Invoke Remove Object Tree
<b>Extends</b>	None
<b>Extension</b>	None
<b>Points</b>	

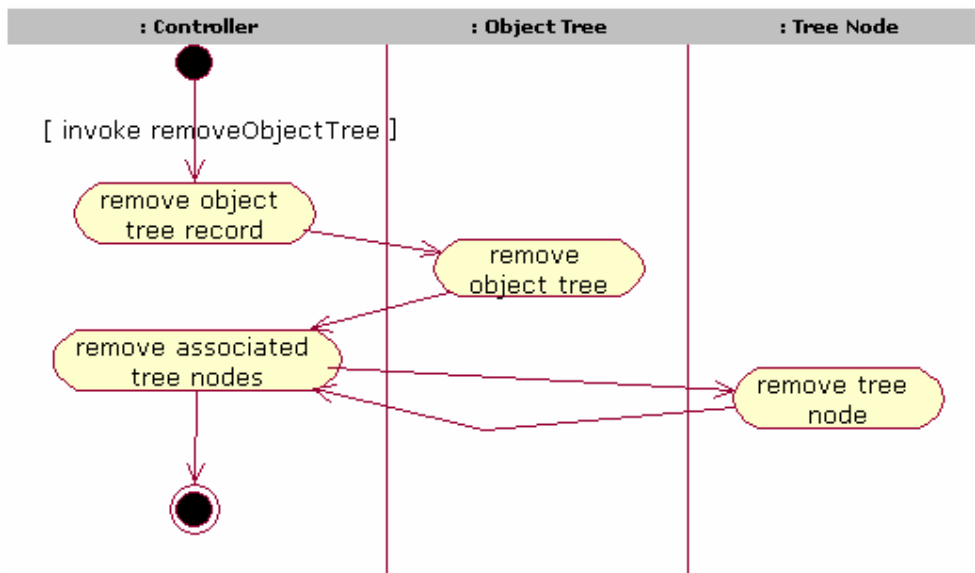


Figure 45 Remove Object Tree Activity Diagram

**3.1.7.3.3.2 Modes of Operation**

This component as the whole system has only one operational mode.

**3.1.7.3.4 L.O.S. Goals**

<b>L.O.S. Requirement:</b>	LR-1: System Dependability - Stable data import/export
<b>Description:</b>	This Component should be able to perform importing/exporting data as many and frequent as desired without crash or unexpected restart.
<b>Measurable:</b>	Component dependability should be measured as the rate of successful importation and exportation over all importation and exportation which is logged on this component’s performance log. And the integrity of data processing (No loss of data)

<b>Relevant:</b>	SR-1: Import usage data from ORACLE-formatted file. SR-2: Remove usage data from the specified log file SR-7: Omit mal-formatted retrieval records. SR-8: Omit usage log file with 0 valid records. OCD 4.4 LS-2: Usability Ensures system usability and therefore satisfaction of the client.
<b>Specific:</b>	Specifies how dependability will be measured

Table 9 LOS goals for Controller component (LR-1)

<b>L.O.S. Requirement:</b>	LR-3: Usability – Maximizing the host resources
<b>Description:</b>	When doing computationally or IO intensive tasks this component should do them in the background allowing normal operation of regular desktop applications: mail, browser, etc.
<b>Measurable:</b>	Estimated by potential user, during performing computationally intensive tasks – how regular desktop applications are usable.
<b>Relevant:</b>	OCD 4.4 LG-2: Usability Ensures system usability and therefore satisfaction of the client.
<b>Specific:</b>	Specifies what applications should be able to run in parallel.

Table 10 LOS goals for Controller component (LR-2)

<b>L.O.S. Requirement:</b>	LR-4: Performance – Organizing data meaningfully for users
<b>Description:</b>	The generated data should be save in a meaning-full organized form to be easily reusable by humans and other applications.
<b>Measurable:</b>	Measured by number of ways data is available to be reused by a person or another applications
<b>Relevant:</b>	SR-5: Graph node statistics OCD 4.4 LG-3: Performance
<b>Specific:</b>	Specifies how meaningfulness of data organization will be measured.

Table 11 LOS goals for Controller component (LR-4)

<b>L.O.S. Requirement:</b>	LR-5: Performance – data of current scale
<b>Description:</b>	This component should be able to perform all of its operations given data input of size of up to 300000 objects.
<b>Measurable:</b>	Successful performing operations on input data of the specified size
<b>Relevant:</b>	SR-3: Relationship generation SR-4: Generate collection structure tree SR-5: Graph node statistics SR-6: Visualization OCD 4.4 LG-3: Performance
<b>Specific:</b>	Specifies acceptable scale limit.

Table 12 LOS goals for Controller component (LR-5)

### 3.1.7.3.5 Constraints

There are no additional constraints.

## 3.1.8 Software Connector Classifiers

No Software Connector Classifiers are defined.

## 3.1.9 Hardware Components

The hardware components identified here is computer. Computer will be used to run the analysis program. The system will first get the raw log data from the local directory which are downloaded by user to their local directory, after the analysis, the result will be stored again.

### 3.1.9.1 Workstation

#### 3.1.9.1.1 Purpose

The purpose this component is to host the proposed system and provide computational and visualization capabilities that are used by the system.

#### 3.1.9.1.2 Classifier

HCC-01

**3.1.9.1.3 L.O.S.**

No Level of Service Goal apply here.

**3.1.10 Hardware Connectors**

No Hardware Connectors are defined.

**3.1.11 Software Components****3.1.11.1 GUI (SC-01)****3.1.11.1.1 Purpose**

To provide interface that allows user to invoke major system capabilities and allows system to output visual information for the user.

**3.1.11.1.2 Classifier**

This component is an instance of SCC-01.

**3.1.11.1.3 L.O.S. Goals**

L.O.S Goals defined for SCC-01 apply here. There are no instance-specific L.O.S. Goals.

**3.1.11.2 Visualizer (SC-02)****3.1.11.2.1 Purpose**

The purpose of this component is to provide 3d hyperbolic interactive view of an Object Tree.

**3.1.11.2.2 Classifier**

This component is an instance of SCC-02.

**3.1.11.2.3 L.O.S. Goals**

L.O.S Goals defined for SCC-02 apply here. There are no instance-specific L.O.S. Goals.

### **3.1.11.3 Controller (SC-03)**

#### **3.1.11.3.1 Purpose**

The purpose of this component is to provide data processing and data management capabilities.

#### **3.1.11.3.2 Classifier**

This component is an instance of SCC-03.

#### **3.1.11.3.3 L.O.S. Goals**

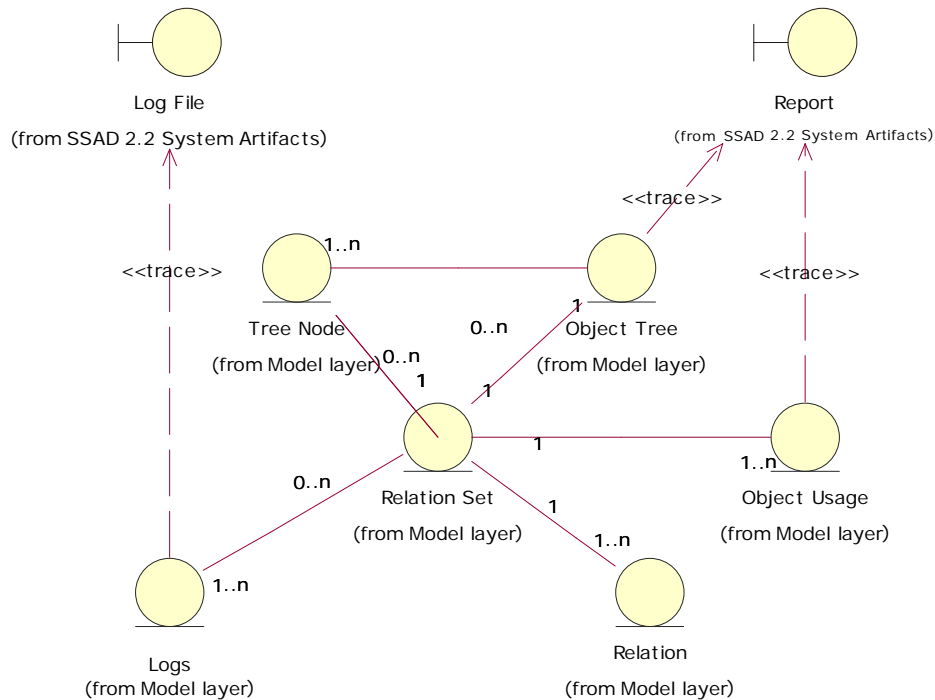
L.O.S Goals defined for SCC-03 apply here. There are no instance-specific L.O.S. Goals.

### **3.1.12 Software Connectors**

No Software Connectors are defined.

## 3.2 Analysis Classes

This section describes Information Classes that the proposed system operates on.



**Figure 46 Information Classes**

This diagram shows relations among Information Classes and artifacts system operates on. System takes Log Files as input, parses them and stores them in Logs database. Input data is analyzed at several levels to produce Relation Sets and then Object Trees.

### 3.2.1 Log File (AC-01)

The USC-ISD has maintained a log file of usage statistic of the USC digital archive. When users visit the Digital archive web site and retrieve certain images, their activities will be logged, the information such as user's IP, item id and time of retrieval are recorded in the log file. These log files will then be imported by user to their local directory to be analyzed by the system. System Capability C-01 [see OCD 4.3] uses this artifact.

### **3.2.2 Logs (AC-02)**

This Information Class represents log data that have been imported from the input Log Files into the system's database for further analysis. Each log record had the following attributes: user ID, session ID, time of access, object ID, source file name. This class is used by the C-01 and C-02 (OCD 4.3).

### **3.2.3 Relation Set (AC-03)**

This class contains information that unites a set of relations that were generated based on available Logs. This class has the following attributes: start date, end date, analysis parameters. This class is used by the C-02 (OCD 4.3).

### **3.2.4 Relation (AC-04)**

This class represents relations that have been generated based on available logs. Relations belong to Relation Sets in a relationship many to one. A relationship has the following attributes: relation set id, objectA, objectB, weight. This class is used by the C-02 (OCD 4.3)

### **3.2.5 Object Usage (AC-05)**

This class represents usage statistics for each object that was collected during generation of a Relationship Set. Thus object usage belongs to Relationship Set in a relation many to one. Object usage has the following attributes: relation set ID, object ID, user ID, and number of accesses. This class is used by C-02 (OCD 4.3)

### **3.2.6 Object Tree (AC-06)**

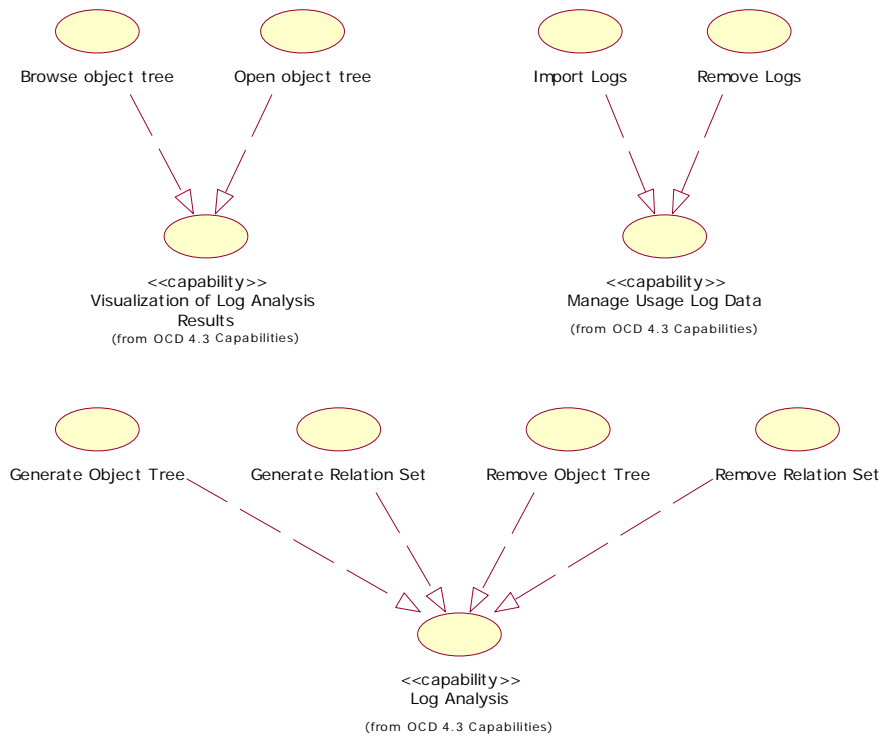
This class represents object tree that is created based on a Relation Set. This Object Tree belongs to Relation Set in a relationship many to one. Object tree represents a set of tree nodes that compose a tree with a following meta-information: relation set ID, analysis parameters. Visualizer component uses this class to construct a 3d hyperbolic view of object tree. This class is used by C-02 and C-03 (OCD 4.3)

### 3.2.7 Tree Node (AC-07)

This class represents information on nodes for different Object Trees. Each Tree Node has the following attributes: object tree ID, order number, tree level, object ID. This class is used by C-02 and C-03 (OCD 4.3)

## 3.3 Behavior

This section will give more detailed description about how the components in section 3.1.4 work together to implement the processes in section 2.3.



**Figure 47 System process implementation**

### 3.3.1.1 Import Logs Implementation

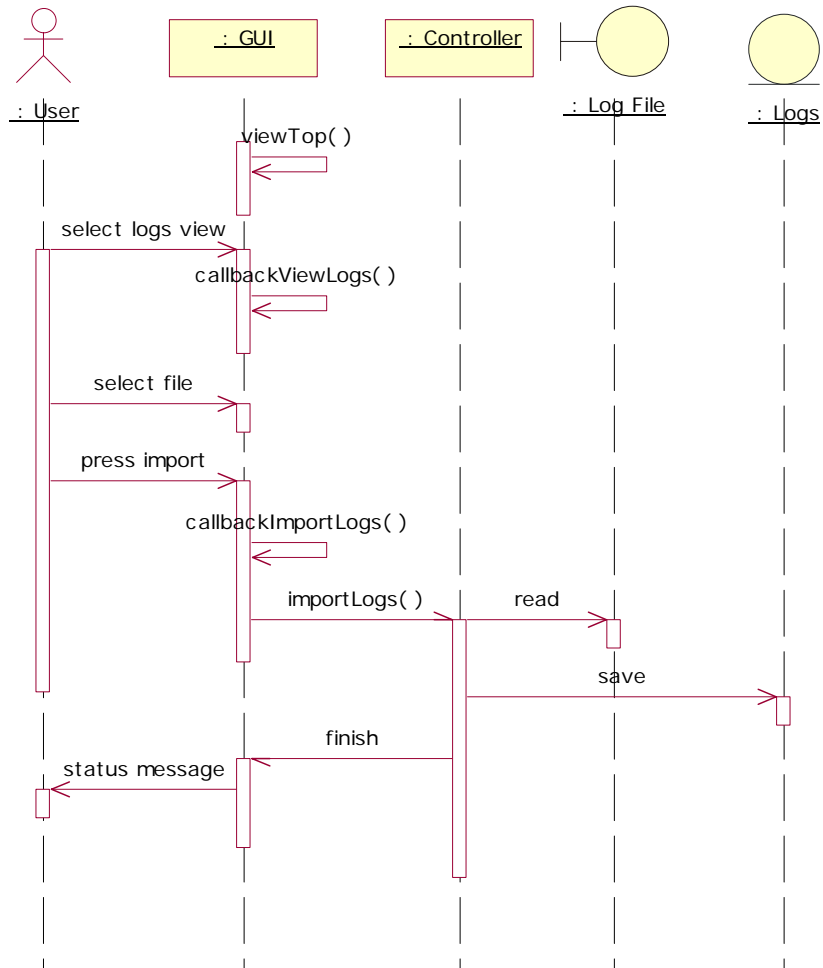


Figure 48 Import Logs Implementation

<b>Identifier</b>	UCR-01
<b>Use-Case Name</b>	Import Logs Implementation
<b>Abstract</b>	No
<b>Purpose</b>	Demonstrates how system imports input data
<b>Actors</b>	User
<b>Importance</b>	Primary
<b>Capability</b>	C-01 (OCD 4.3)
<b>Requirements</b>	SSRD. 3.2.1 (SR-1, SR-2) 4.1.1(.IR-1)
<b>Risks</b>	None
<b>High-Risk?</b>	No

<b>Architecturally Significant?</b>	No
<b>Development Status</b>	LCA
<b>Overview</b>	
<b>User Interface</b>	See OCD 5 - prototype
<b>Pre-conditions</b>	Log file exists.
<b>Post-conditions</b>	Log file imported/deleted.
<b>Specializes</b>	None
<b>Includes</b>	None
<b>Extends</b>	None
<b>Extension Points</b>	None

### 3.3.1.2 Remove Logs Implementation

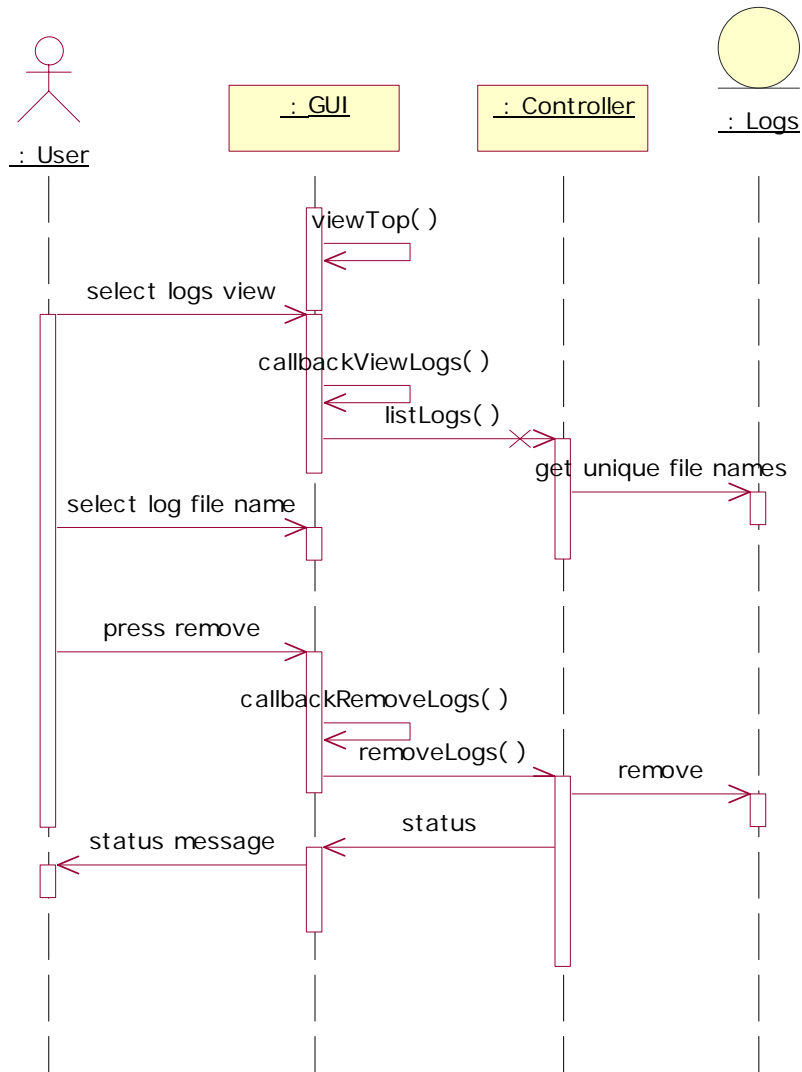


Figure 49 Remove Logs Implementation

### 3.3.1.3 Generate Relation Set Implementation

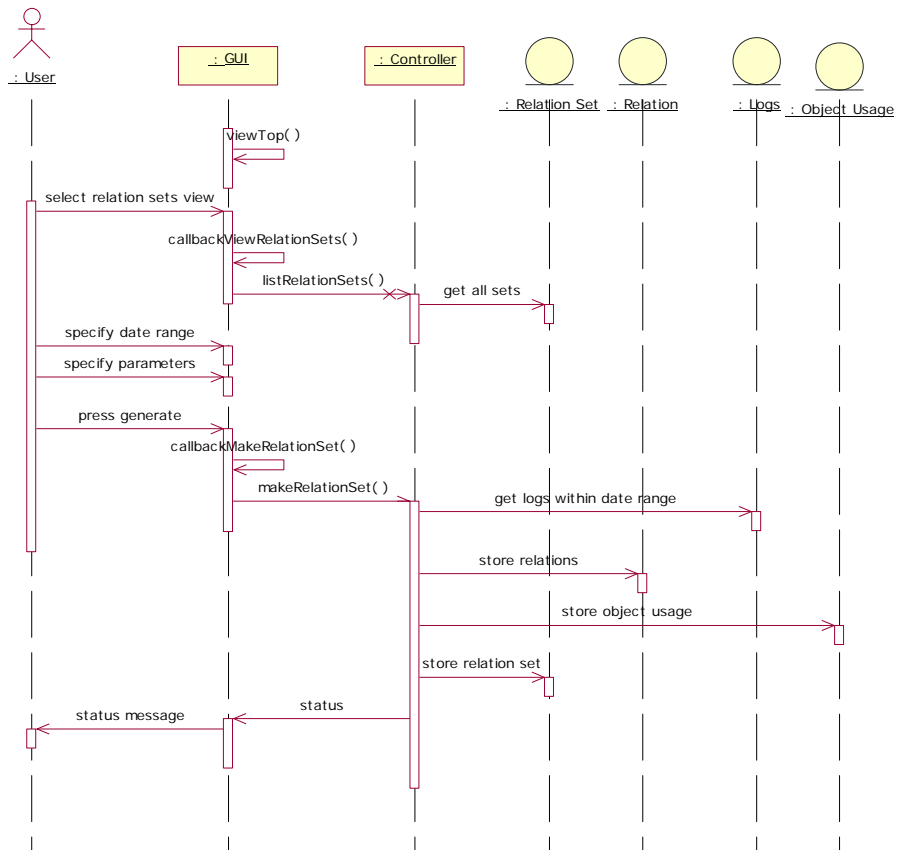


Figure 50 Generate Relation Set Implementation

### 3.3.1.4 Remove Relation Set Implementation

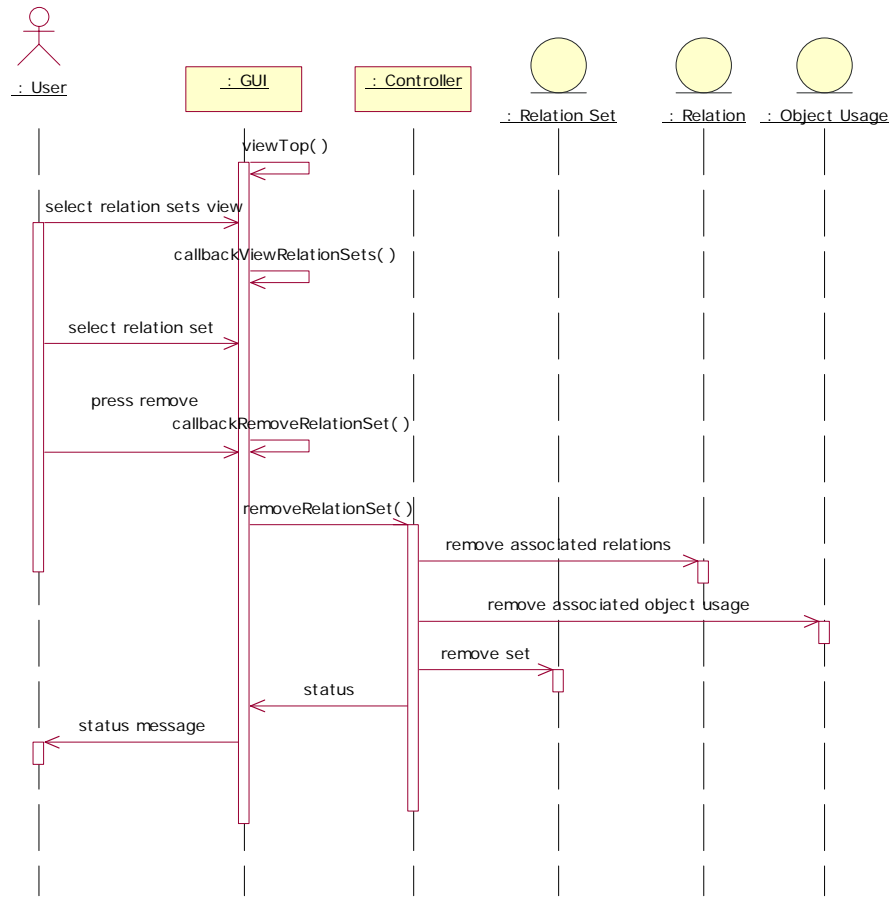


Figure 51 Remove Relation Set Implementation

### 3.3.1.5 Generate Object Tree Implementation

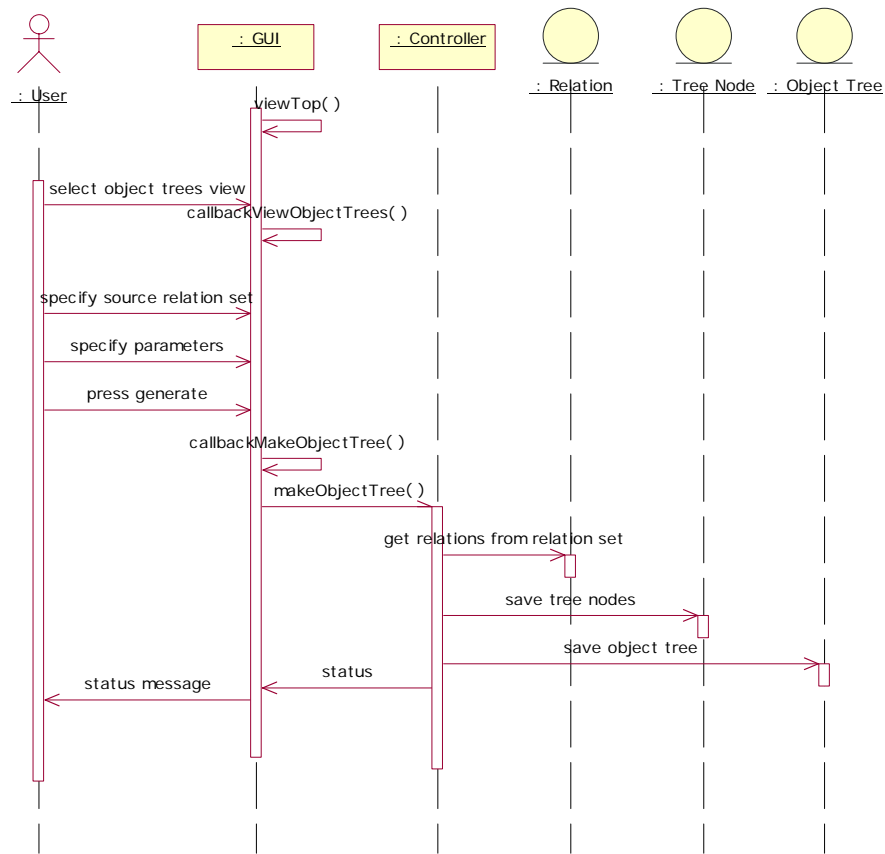


Figure 52 Generate Object Tree Implementation

### 3.3.1.6 Remove Object Tree Implementation

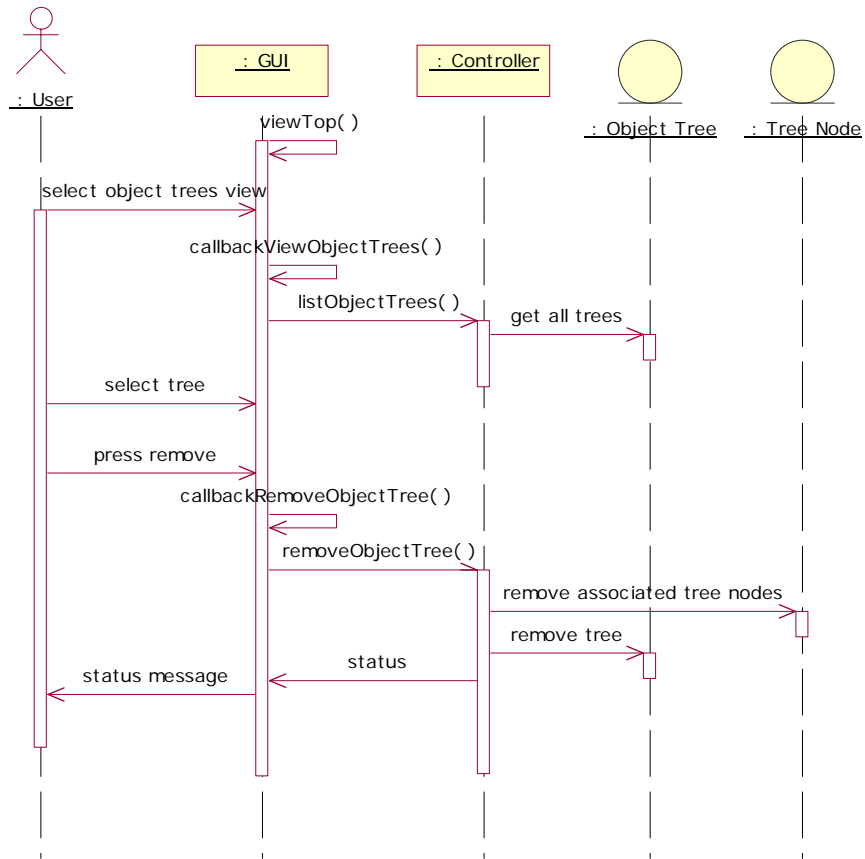
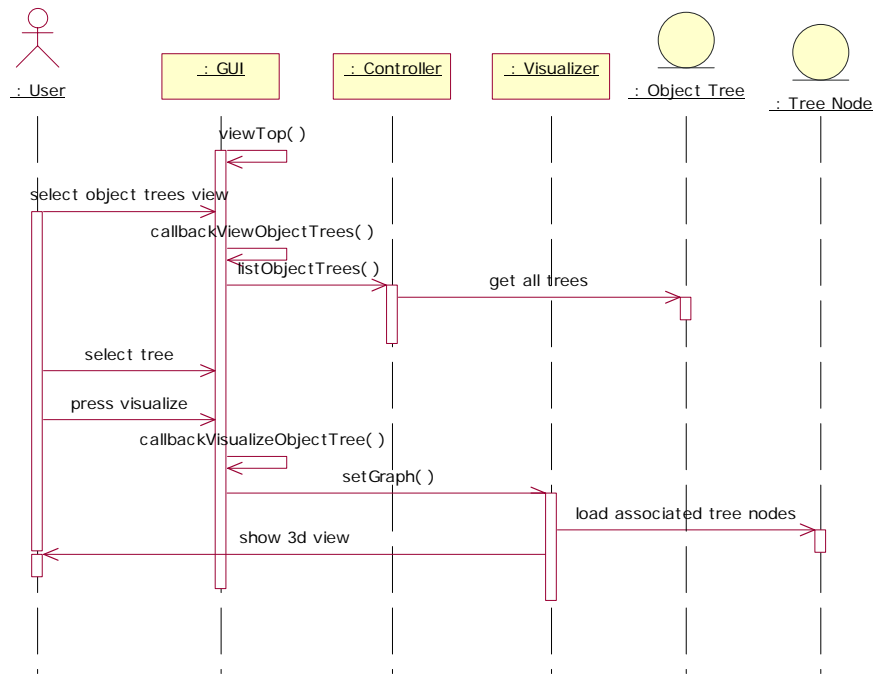


Figure 53 Remove Object Tree Implementation

### 3.3.1.7 Visualize Object Tree Implementation



### 3.3.1.8 Browse Object Tree Implementation

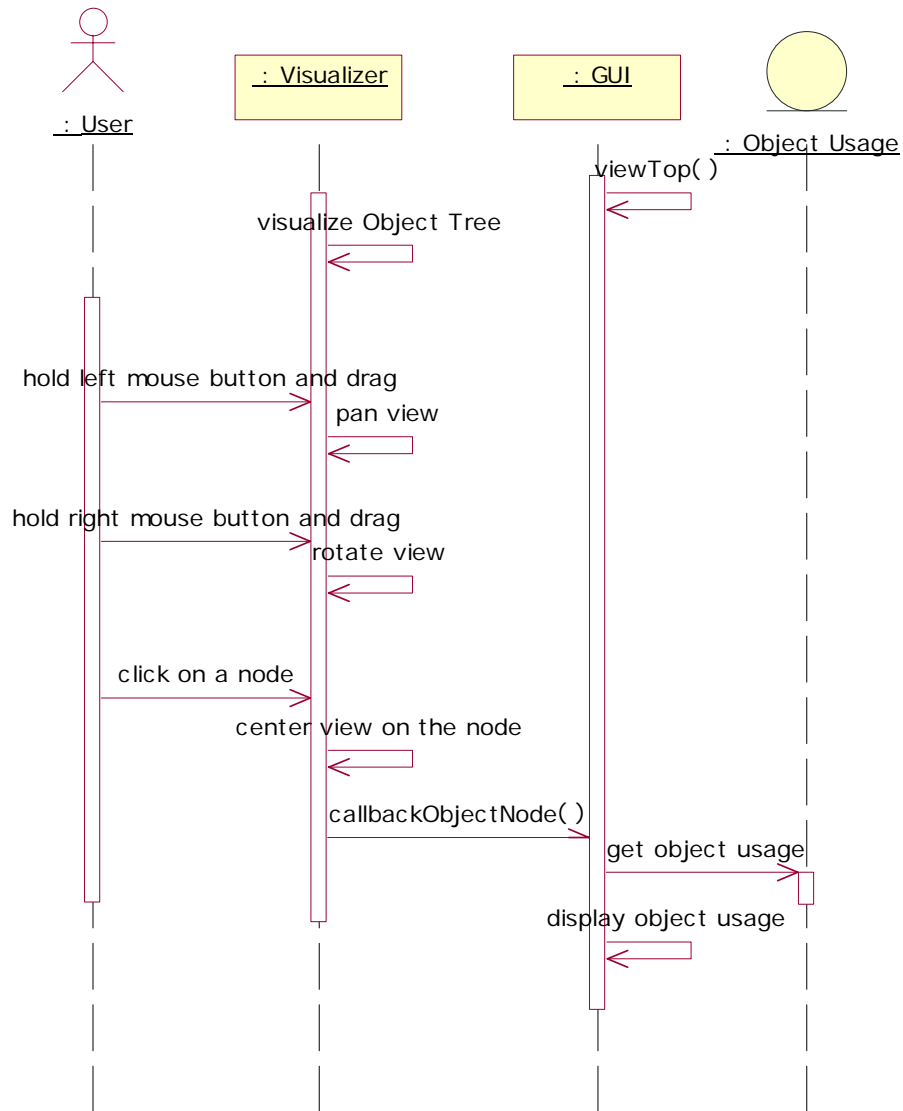


Figure 54 Browse Object Tree Implementation

## 3.4 L.O.S Projected

This section describes the elements of the architecture to which each system L.O.S. applies and the projected value of each system L.O.S.

L.O.S Goal	Applies To	How	Projected Value	Evaluation Technique
LR-1 [SSRD 5]: Dependability	SCO-01, SCO -02, SCO -03, SCO -04	Equally	Pass 95% of test cases	Estimation
LR-2 [SSRD 5]: Usability	SCO -04	Equally	Satisfied with the client	Estimation
LR-3 [SSRD 5]: Operability in multitasking environment	SCO -03	Equally	Pass 95% of test cases	Estimation
LR-4[SSRD 5]: Performance on data of current scale	SCO -04	Equally	Able to perform properly with 10% increase.	Estimation

Table 13 L.O.S projected

### 3.5 Architectural Styles, Patterns & Frameworks

Our system's architecture is defined in terms of Model-View-Controller design pattern which allows for the following benefits

Name	Description	Benefits, Cots & limitations
Multi-layer Pattern	Multi-Layer system is a kind of software system and uses multi-layer architecture pattern. See <a href="http://www.site.uottawa.ca:4321/oose/index.html#multi-layer_table">http://www.site.uottawa.ca:4321/oose/index.html#multi-layer_table</a>	It replaces a layer by an improved version, or by one of different capabilities. In addition, it also increases reusability. It divides and conquers since the separate layers can be independently designed.

Table 14 Architectural Styles, Patterns &amp; Frameworks

## 4. Implementation Design

In this section we will design a technology-specific implementation for the system by refining the general architecture defined during Architecture Design & Analysis (SSAD 3).

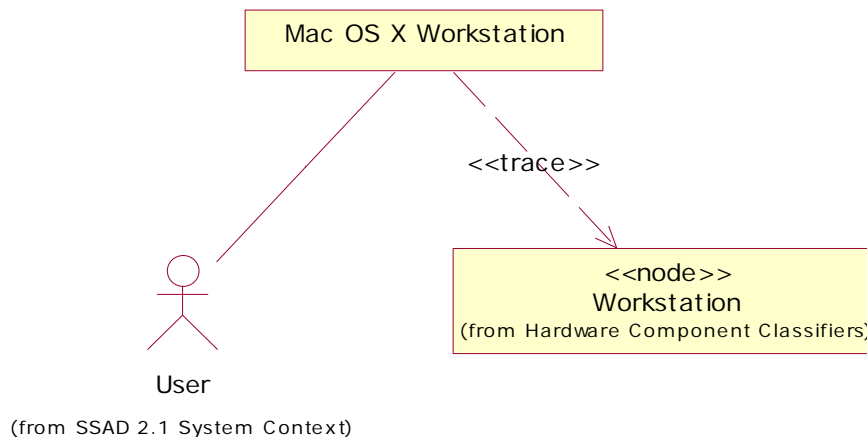
### 4.1 Structure

#### 4.1.1 Topology

No changes in system topology since section 3.1.1.

#### 4.1.2 Hardware Classifier Model

This section describes implementation specific hardware components that will be used by the system.

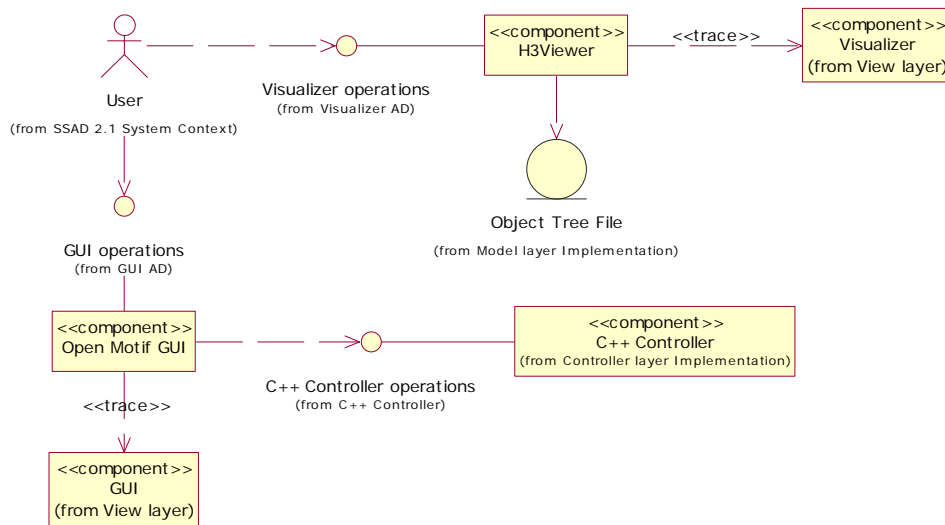


**Table 15 Hardware Classifier Implementation**

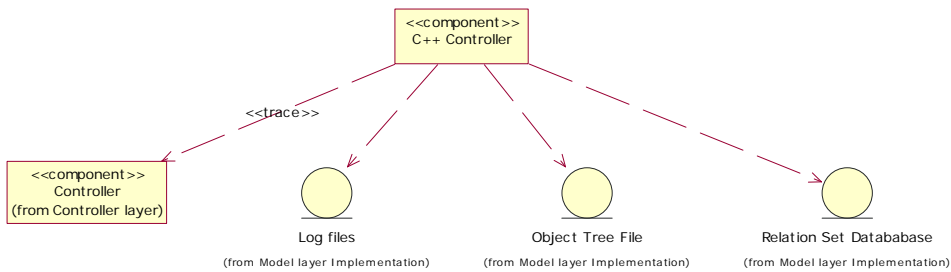
#### 4.1.3 Software Classifier Model

This section describes implementation specific Software Classifier Model. According to the system topology components are separated into three layers. The following

diagrams show implementation specific configuration of components in each layer and also mapping of components to generic software classifiers defined in 3.1.3.



**Figure 55 View layer Implementation**



**Figure 56 Controller layer Implementation**

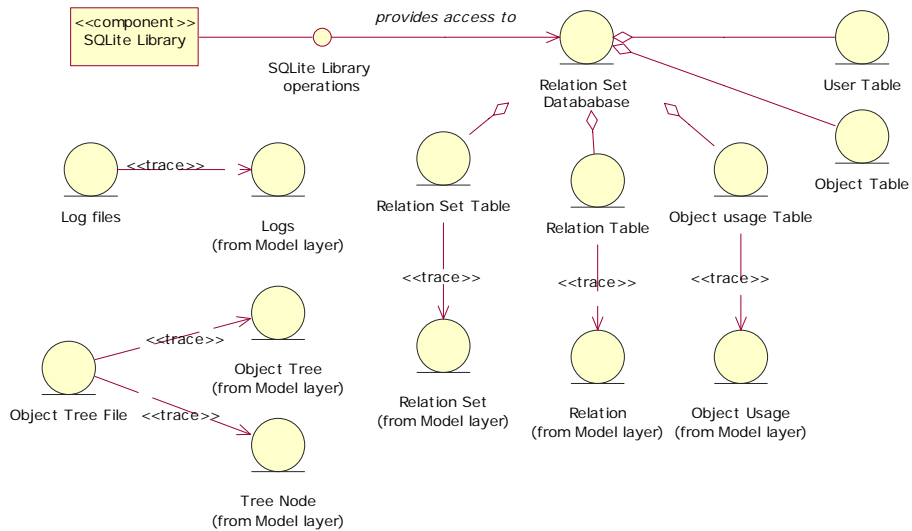


Figure 57 Model layer Implementation

### 4.1.4 Deployment Model

The following diagram displays configuration hardware and software component instances that constitutes a working system

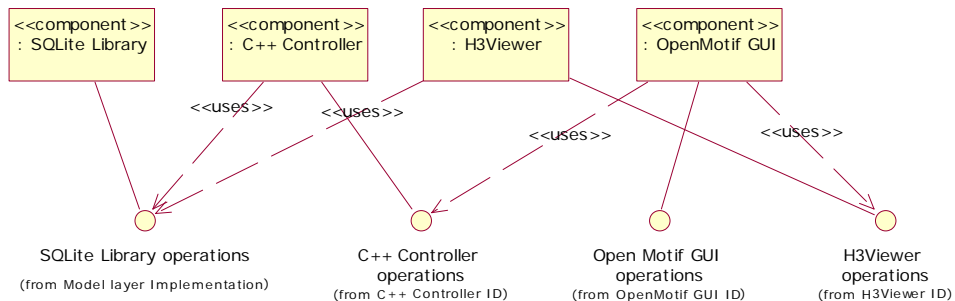


Figure 58 Deployment Model

### 4.1.5 Hardware Component Classifiers

#### 4.1.5.1 Mac OS X Workstation (HCC-01)

##### 4.1.5.1.1 Purpose

The purpose of this component is to host the Usage Log Analysis System and provide it with computational and visualization resources. As a result of negotiations with the

client Apple computer hardware and Mac OS X operating system was chosen as target platform for the proposed system.

#### **4.1.5.1.2 L.O.S. Goals**

See 3.1.5.1.2.

### **4.1.6 Hardware Connector Classifiers**

See 3.1.6.

### **4.1.7 Software Component Classifiers**

#### **4.1.7.1 Open Motif GUI (SCC-01)**

##### **4.1.7.1.1 Purpose**

Purpose of this component is to provide interface for user to be able to invoke analysis and visualization capabilities of the proposed system.

##### **4.1.7.1.2 Interface**

###### **4.1.7.1.2.1 callbackRelationSetForm**

Parameters: None

Preconditions: Main form was properly initialized and displayed. User selected option to generate a new Relation Set.

Postconditions: Form to generate new Relation Set is initialized and presented to the user.

###### **4.1.7.1.2.2 callbackObjectTreeForm**

Parameters: None

Preconditions: Main form was properly initialized and displayed. User selected option to create a new Tree.

Postconditions: New Object Tree Form is initialized and displayed.

###### **4.1.7.1.2.3 callbackViewObjectTree**

Parameters: None

Preconditions: Main form was properly initialized and displayed. User selected option to view Object Tree.

Postconditions: Browse file dialog is presented to the user to select an Object Tree file to be visualized. Setup a handler for file selection event to call setGraph operation of H3Viewer component.

#### **4.1.7.1.2.4 callbackAddLogFile**

Parameters: Path to a source Log File.

Preconditions: NewRelationSet Form was properly initialized and displayed. User clicked on browse button selected a source log file and clicked select button.

Postconditions: The log file that user specified is added to the List of source Log Files of the New Relation Set Form.

#### **4.1.7.1.2.5 callbackRemoveLogFile**

Parameters: List number of the Log File.

Preconditions: New Relation Set Form was properly initialized and displayed.

Specified Log File was previously selected and added to the list by the user. User selected Log File in the list and pressed remove button.

Postconditions: Specified Log File is no longer contained in the list.

#### **4.1.7.1.2.6 callbackMakeRelationSet**

Parameters: Values of the filled New Relation Set Form

Preconditions: New Relation Set Form was properly initialized and displayed. User filled the form and pressed generate button.

Postconditions: makeRelationSet operation of the Controller component is called with the corresponding parameters taken from the filled form.

#### **4.1.7.1.2.7 callbackMakeTree**

Parameters: Values of the filled New Object Tree Form

Preconditions: New Object Tree Form was properly initialized and displayed. User filled the form and pressed generate button.

Postconditions: makeObjectTree operation of the Controller component is called with the corresponding parameters.

#### **4.1.7.1.3 Parameters**

This component has no parameters defined.

#### 4.1.7.1.4 Behavior

##### 4.1.7.1.4.1 Processes

Processes are similar to ones defined in 3.1.7.1.4.1

##### 4.1.7.1.4.2 Modes of Operation

One operational mode.

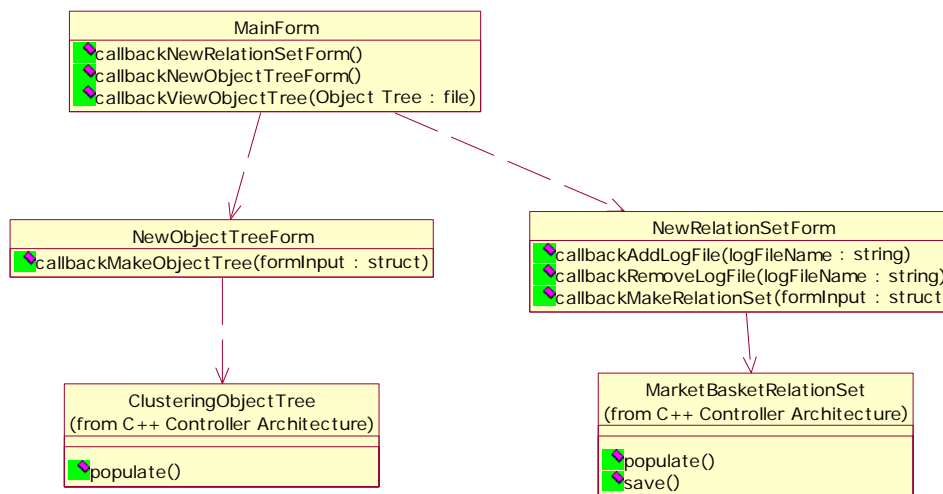
##### 4.1.7.1.5 L.O.S. Goals

See 3.1.7.1.4

##### 4.1.7.1.6 Constraints

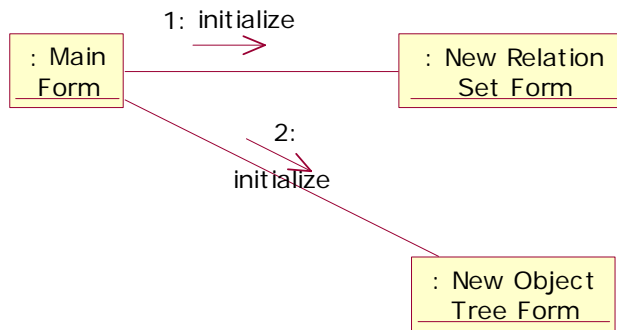
See 3.1.7.1.5

##### 4.1.7.1.7 Internal Architecture



**Figure 59 Open Motif GUI Class Diagram**

This component composed of three classes, which provide operations that combined together comprise the interface for this component. The three classes are forms which provide necessary GUI for corresponding operations. The classes will be described in detail later.



**Figure 60 Openotif GUI Collaboration Diagram**

### 4.1.7.2 H3Viewer (SCC-02)

#### 4.1.7.2.1 Purpose

Purpose of this component is to provide 3d interactive visualization capability that allows user to examine results of analysis process.

#### 4.1.7.2.2 Interface

##### 4.1.7.2.2.1 setGraph

This function initializes 3d hyperbolic view with the specified source Object Tree  
Parameters: path to Object Tree file to be visualized

Preconditions: H3Viewer Frame was properly initialized and displayed. Source Object Tree file is available on local machine.

Postconditions: 3d hyperbolic view of the specified Object Tree is presented to the user.

##### 4.1.7.2.2.2 callbackSelectObject

Parameters: Object id that was selected

Preconditions: H3Viewer frame was properly initialized. setGraph was called and valid Object Tree was loaded. User clicked on an Object in the 3d view frame.

Postconditions: View is centered on selected Obejct. callbackShowObjectInfo operation of this component is called.

##### 4.1.7.2.2.3 callbackShowObjectInfo

Parameters: Obejct ID

Preconditions: H3Viewer frame was properly initialized. Valid Object Tree was loaded.

Postconditions: Usage information for the object and information on items contained in the object is displayed in Object frame.

#### **4.1.7.2.3 Parameters**

No parameters defined for this component

#### **4.1.7.2.4 Behavior**

##### **4.1.7.2.4.1 Processes**

Processes are similar to ones defined in 3.1.7.2.3.1

##### **4.1.7.2.4.2 Modes of Operation**

One Operational Mode

#### **4.1.7.2.5 L.O.S. Goals**

See 3.1.7.2.4

#### **4.1.7.2.6 Constraints**

No constraints are defined.

#### **4.1.7.2.7 Internal Architecture**

This component is composed of two frame classes. H3Viewer frame provides 3d hyperbolic view of the Object Tree and Object Info frame provides view of the Object Usage and other related information.

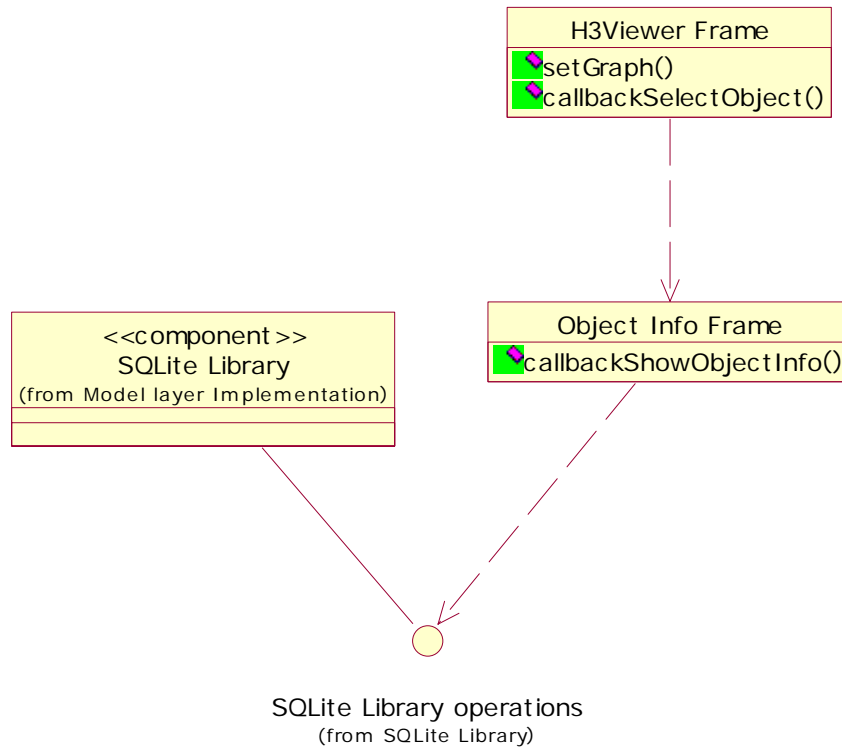


Figure 61 H3Viewer Class Diagram

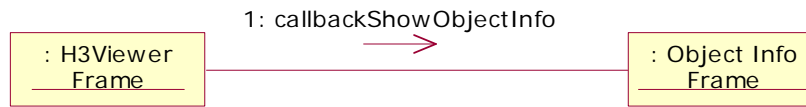


Figure 62 H3Viewer Collaboration Diagram

### 4.1.7.3 C++ Controller (SCC-03)

#### 4.1.7.3.1 Purpose

Purpose of this component is to provide major data processing and analysis capabilities of the system.

#### 4.1.7.3.2 Interface

##### 4.1.7.3.2.1 makeRelationSet

Parameters: List of paths to source Log Files. Relations Set options.

Preconditions: Target source Log Files are present on local machine.

Postconditions: Relation Set is generated according to the following algorithm:

For each user session, find all unique items that were accessed, increment bi-directional relationships between each pair of accessed items by one. After all data is processed – normalize weights of relationships, dividing by minimum weight.

#### **4.1.7.3.2.2 saveRelationSet**

Parameters: Path to store Relation Set database file. Potiner to generated Relation Set.

Preconditions: RelationSet was properly generated

Postconditions: RelationSet is saved at the specified location in form of SQLite database.

#### **4.1.7.3.2.3 makeTree**

Parameters: Path to source Relation Set database. Object Tree parameters

Preconditions: Specified Relation Set is available in specified location.

Postconditions: Object Tree is generated according to the following recursive algorithm: For a given Relation Set: determine the most central item – the item with highest sum of incoming and outgoing links. Make that item root of this level. Cluster the relation set and make the resulting sub sets – children of the current root. Repeat the algorithm for each of sub sets.

#### **4.1.7.3.2.4 saveTree**

Parameters: Destination path where Object Tree will be saved.

Precondition: Object Tree was properly generated

Postconditions: Object Tree is saved in the specified location in the format that is supported by the H3Viewer.

#### **4.1.7.3.3 Parameters**

No Parameters defined

#### **4.1.7.3.4 Behavior**

##### **4.1.7.3.4.1 Processes**

See 3.1.7.3.3

##### **4.1.7.3.4.2 Modes of Operation**

One operational mode.

**4.1.7.3.5 L.O.S. Goals**

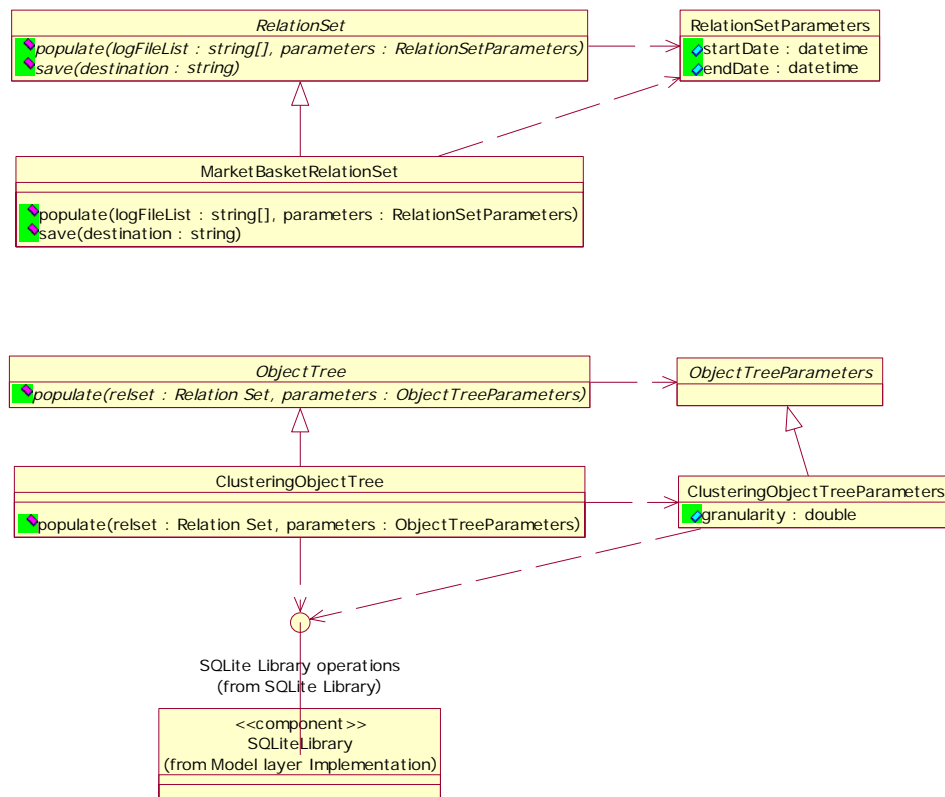
See 3.1.7.3.4

**4.1.7.3.6 Constraints**

No constraints defined

**4.1.7.3.7 Internal Architecture**

Functionality of this component will be implemented by the following two classes: MarketBasketRelationSet that extends abstract class RelationSet and ClusteringObjectTree that extends abstract class ObjectTree. Both of those classes have operations called populate that take different parameters: RelationSetParameters and ObjectTreeParameters.



**Figure 63 C++ Controller Class Diagram**

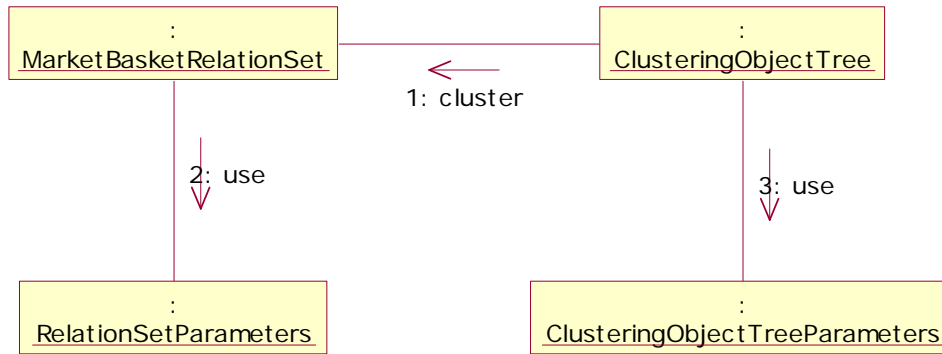


Figure 64 C++ Controller Collaboration Diagram

#### 4.1.7.4 SQLite Library (SCC-04)

##### 4.1.7.4.1 Purpose

Purpose of this component is to provide functionality that allows storing and querying arbitrary data using SQL syntax.

##### 4.1.7.4.2 Interface

###### 4.1.7.4.2.1 sqlite\_open

Parameters: Path to the target database

Preconditions: Database file is properly initialized and available on the local machine

Postconditions: Specified database file is open and pointer to the connection is returned.

###### 4.1.7.4.2.2 sqlite\_exec

Parameters: Pointer to the database connection. SQL query.

Preconditions: Database file was open and connection established.

Postconditions: Specified SQL query was executed on the specified database.

###### 4.1.7.4.2.3 sqlite\_close

Parameters: Pointer to open database connection.

Preconditions: Specified database connection was open.

Postconditions: Specified database connection is closed.

##### 4.1.7.4.3 Parameters

No parameters defined for this class.

#### **4.1.7.4.4 Behavior**

##### **4.1.7.4.4.1 Processes**

Processes for this component are defined in detail in third party documentation.

<http://www.sqlite.org/docs.html>

##### **4.1.7.4.4.2 Modes of Operation**

One operational mode.

##### **4.1.7.4.5 L.O.S. Goals**

L.O.S goals for this component are defined in third party documentation.

<http://www.sqlite.org/docs.html>

##### **4.1.7.4.6 Constraints**

No constraints defined for this component.

##### **4.1.7.4.7 Internal Architecture**

Internal Architecture for this component is defined in third party documentation.

<http://www.sqlite.org/docs.html>

### **4.1.8 Hardware Components**

#### **4.1.8.1 Client Mac OS X Workstation**

##### **4.1.8.1.1 Purpose**

The purpose of this component is to serve as a host for the proposed Usage Log Analysis System and to provide computational and visualization hardware capabilities.

##### **4.1.8.1.2 Classifier**

This hardware component is an instance of the classifier computer HCC-01

##### **4.1.8.1.3 L.O.S.**

L.O.S. goals defined for hardware component classifier HCC-01 apply here.

## **4.1.9 Hardware Connectors**

No hardware Connectors defined.

## **4.1.10 Software Components**

### **4.1.10.1 Open Motif GUI (SC-01)**

#### **4.1.10.1.1 Purpose**

This is implementation of user interface functionality defined in 3.1.11.1 using open source windowing toolkit – Open Motif.

#### **4.1.10.1.2 Classifier**

This component is an instance of SCC-01.

#### **4.1.10.1.3 L.O.S. Goals**

L.O.S Goals defined for SCC-01 apply here. There are no instance-specific L.O.S. Goals.

### **4.1.10.2 H3Viewer (SC-02)**

#### **4.1.10.2.1 Purpose**

The purpose of this component is to provide 3d hyperbolic interactive view of an Object Tree. This functionality is implemented using open source graph layout and visualization library H3Viewer and interface toolkit API provided by Open Motif.

#### **4.1.10.2.2 Classifier**

This component is an instance of SCC-02.

#### **4.1.10.2.3 L.O.S. Goals**

L.O.S Goals defined for SCC-02 apply here. There are no instance-specific L.O.S. Goals.

### **4.1.10.3 C++ Controller (SC-03)**

#### **4.1.10.3.1 Purpose**

This component is a C++ implemen

#### **4.1.10.3.2 Classifier**

This component is an instance of SCC-03.

#### **4.1.10.3.3 L.O.S. Goals**

L.O.S Goals defined for SCC-03 apply here. There are no instance-specific L.O.S. Goals.

### **4.1.10.4 SQLite Library (SC-04)**

#### **4.1.10.4.1 Purpose**

Provide operations that allow storing and querying data using SQL syntax.

#### **4.1.10.4.2 Classifier**

This component is an instance of SCC-03.

#### **4.1.10.4.3 L.O.S. Goals**

L.O.S Goals defined for SCC-03 apply here. There are no instance-specific L.O.S. Goals.

## **4.1.11 Software Connectors**

No connectors are defined for the proposed system.

## **4.1.12 Implementation Classes**

### **4.1.12.1 MainForm**

#### **4.1.12.1.1 Purpose**

Purpose of this class is to initialize and display the Top view in the proposed system. This form will provide interface to select available sub-views and to invoke system

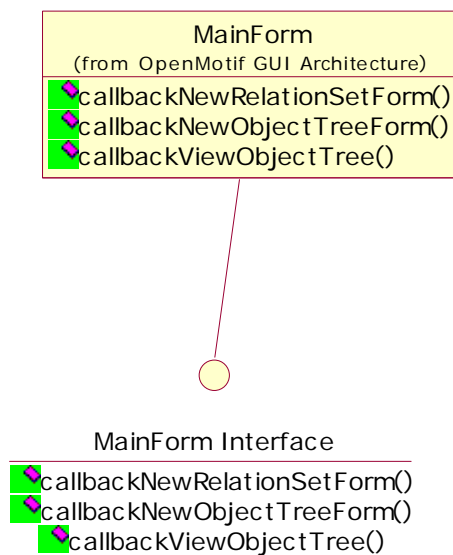
functionality it will handle the select events and will call corresponding operations of other classes.

#### 4.1.12.1.2 Defined In: SC-01

Open Motif GUI.

#### 4.1.12.1.3 Interface

See 4.1.7.1.2.1, 4.1.7.1.2.2, 4.1.7.1.2.3



**Figure 65 MainForm Interface**

#### 4.1.12.1.4 Parameters

No Parameters defined

#### 4.1.12.1.5 Attributes

No attributes defined

#### 4.1.12.1.6 Operations

##### 4.1.12.1.6.1 callbackNewRelationSetForm (OP-01)

Identifier: OP-01

Operation name: callbackNewRelationSetForm

Parameters: None

Result: None

Purpose: To allow user to select sub-view for generating new relation sets

Pre-conditions: Main Form is properly initialized and displayed

Post-conditions: NewRelationSetForm is initialized and displayed

Visibility: public

Abstract: No

Method: uses OpenMotif API to initialize new form

#### **4.1.12.1.6.2 callbackNewObjectTreeForm (OP-02)**

Identifier: OP-02

Operation name: callbackNewObjectTreeForm

Parameters: None

Result: None

Purpose: allow user to select sub-view for generating new Object Trees

Pre-conditions: MainForm properly initialized

Post-conditions: NewObjectTreeForm is initialized and displayed

Visibility: public

Abstract: No

Method: uses OpenMotif API to initialize new form

#### **4.1.12.1.6.3 callbackViewObjectTree (OP-03)**

Identifier: OP-03

Operation name: callbackViewObjectTree

Parameters: return value of the browse file form for selecting source Object Tree file

Result: None

Purpose: Handle even of user selecting option to visualize object tree.

Pre-conditions: MainForm was properly initialized and displayed. Target Object Tree was generate and is available from local disk.

Post-conditions: Opens separate window with 3d hyperbolic view of the specified tree and additional frame for displaying ObjectInfo Visibility: public

Abstract: No

Method: uses OpenMotif API to initialize new form

#### **4.1.12.1.7 State Behavior**

Class operates in one state.

**4.1.12.1.8 L.O.S.**

See 4.1.10.1.3

**4.1.12.1.9 Constraints**

No constraints defined

**4.1.12.2 NewRelationSetForm****4.1.12.2.1 Purpose**

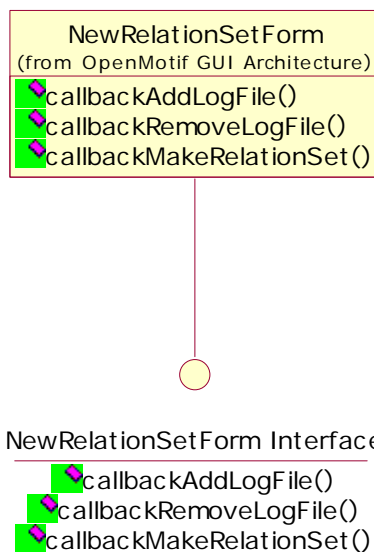
Purpose of this class is to initialize and display interface for generating Relation Sets.

**4.1.12.2.2 Defined In: SC-01**

Defined in Open Motif GUI

**4.1.12.2.3 Interface**

See 4.1.7.1.2.4, 4.1.7.1.2.5, 4.1.7.1.2.6



**Figure 66 NewRelationSetForm Interface**

**4.1.12.2.4 Parameters**

No parameters defined

#### **4.1.12.2.5 Attributes**

No attributes defined

#### **4.1.12.2.6 Operations**

##### **4.1.12.2.6.1 callbackAddLogFile (OP-04)**

Identifier: OP-04

Operation name: callbackAddLogFile

Parameters: string of Log File name

Result: None

Purpose: Allows user to add multiple source Log Files for generation of a RelationSet

Pre-conditions: NewRelationSetForm was properly generated. User selected a Log File using file browsing menu and pressed Add button.

Post-conditions: The specified LogFile is added to the list of source Log Files on the NewRelationSetForm . List of source LogFiles is updated accordingly

Visibility: public

Abstract: No

Method: Uses standard file browsing and selection widget from OpenMotif API to allow user to find and select source files. Uses scrollable list widget to present selected files.

##### **4.1.12.2.6.2 callbackRemoveLogFile (OP-05)**

Identifier: OP-05

Operation name: callbackRemoveLogFile

Parameters: Number of the selected Log File in the list

Result: None

Purpose: Allows user to remove unwanted source Log Files from the list.

Pre-conditions: NewRelationSetForm is properly initialized. User selected Log File from the list of added Log Files and pressed remove

Post-conditions: Selected Log File is removed from the list

Visibility: Public

Abstract: No

Method: Uses scrollable selectable list widget to present list of added LogFiles.

##### **4.1.12.2.6.3 callbackMakeRelationSet (OP-06)**

Identifier: OP-06

Operation name: callbackMakeRelationSet

Parameters: Pointer to the filled NewRelationSetForm

Result: None

Purpose: Handles the event of user submitting NewRelationSetForm

Pre-conditions: NewRelationSetForm was properly initialized and displayed. User filled the form and submitted it.

Post-conditions: MarketBasketRelationSet is created and its operation “populate” is called.

Visibility: Public

Abstract: No

Method: Standard C++ syntax for creating an object and calling it’s methods.

#### **4.1.12.2.7 State Behavior**

No significant state behavior defined.

#### **4.1.12.2.8 L.O.S.**

See 4.1.10.1.3

#### **4.1.12.2.9 Constraints**

No constraints defined.

### **4.1.12.3 NewObjectTreeForm**

#### **4.1.12.3.1 Purpose**

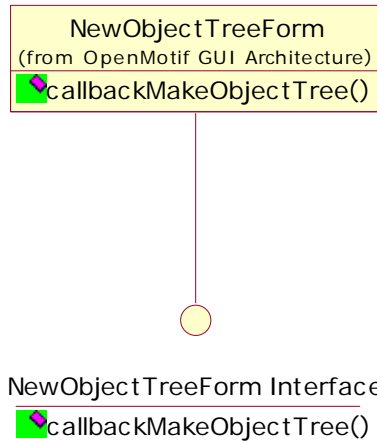
The purpose of this class is to provide interface for creating Object Trees.

#### **4.1.12.3.2 Defined In: SC-01**

Open Motif GUI

#### **4.1.12.3.3 Interface**

See 4.1.7.1.2.7



**Figure 67 NewObjectTreeForm Interface**

#### 4.1.12.3.4 Parameters

No parameters defined.

#### 4.1.12.3.5 Attributes

No attributes defined.

#### 4.1.12.3.6 Operations

##### 4.1.12.3.6.1 callbackMakeObjectTree (OP-07)

Identifier: OP-07

Operation name: callbackMakeObjectTree

Parameters: Pointer to filled NewObejctTreeForm

Result: None

Purpose: Handles event of user submitting the NewObjectTreeForm.

Pre-conditions: NewObjectTreeForm was properly initialized and displayed. User pressed submit button on the form.

Post-conditions: Instance of ClusteringObjectTree is created and its operation populate is called

Visibility: Public

Abstract: No

Method: standard C++

#### 4.1.12.3.7 State Behavior

No significant state behaviors defined

**4.1.12.3.8 L.O.S.**

See 4.1.10.1.3

**4.1.12.3.9 Constraints**

No constraints defined.

**4.1.12.4 H3ViewerFrame****4.1.12.4.1 Purpose**

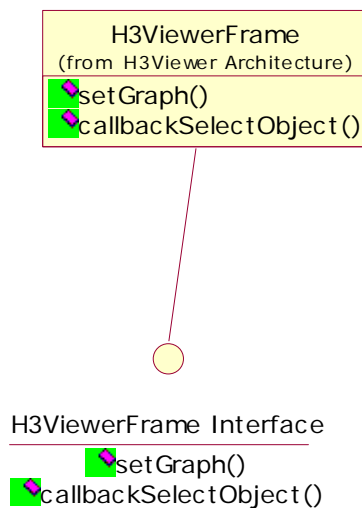
Provide interactive 3d hyperbolic view of object tree.

**4.1.12.4.2 Defined In: SC-02**

H3Viewer Software Component

**4.1.12.4.3 Interface**

See 4.1.7.2.2.1, 4.1.7.2.2.2



**Figure 68 H3ViewerFrame Interface**

**4.1.12.4.4 Parameters**

No parameters defined

#### **4.1.12.4.5 Attributes**

No attributes defined

#### **4.1.12.4.6 Operations**

##### **4.1.12.4.6.1 setGraph (OP-08)**

Identifier: OP-08

Operation name: setGraph

Parameters: Path to the file which contains representation of the Object Tree in compatible format

Result: None

Purpose: Provides visualization capability

Pre-conditions: H3ViewerFrame is properly initialized

Post-conditions: The specified ObjectTree is displayed in 3d hyperbolic view

Visibility: public

Abstract: No

Method: Implementation is done using third party software component H3Viewer

##### **4.1.12.4.6.2 callbackSelectObject (OP-09)**

Identifier: OP-09

Operation name: callbackSelectObject

Parameters: Object ID

Result: None

Purpose: Handles the event of user selecting an object

Pre-conditions: H3Viewer frame was properly initialized and displayed

Post-conditions: 3d view is rotated and panned such that the selected object node is in the center of the view

Visibility: public

Abstract: No

Method: Using H3Viewer API.

#### **4.1.12.4.7 State Behavior**

No significant state behavior

**4.1.12.4.8 L.O.S.**

See 4.1.7.2.5

**4.1.12.4.9 Constraints**

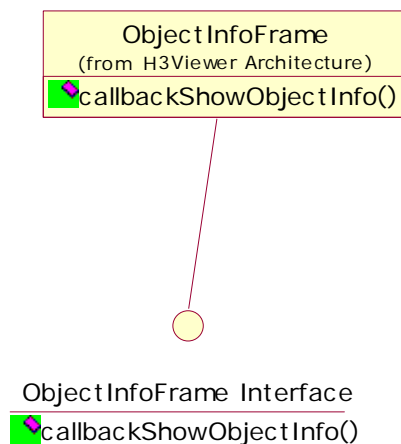
No constraints defined.

**4.1.12.5 ObjectInfoFrame****4.1.12.5.1 Purpose**

Purpose of this class is to provide interface for displaying Object Usage data and other Object information.

**4.1.12.5.2 Defined In: SC-02****4.1.12.5.3 Interface**

See 4.1.7.2.2.3



**Figure 69 ObjectInfoFrame Interface**

**4.1.12.5.4 Parameters****4.1.12.5.4.1 RelationSetID**

This parameter should be passed by the MainForm class when instantiating this class. The RelationSetID should be stored as a comment in the header of the ObjectTree file.

**4.1.12.5.5 Attributes**

No attributes defined

#### **4.1.12.5.6 Operations**

##### **4.1.12.5.6.1 callbackShowObjectInfo (OP-10)**

Identifier: OP-10

Operation name: callbackShowObjectInfo

Parameters: Object ID

Result: None

Purpose: Handles event of user selecting an object in the 3d hyperbolic view.

Pre-conditions: H3Viewer and ObjectInfoFrame were properly initialized and displayed. User selected an Object in the H3Viewer frame

Post-conditions: Usage information for the selected object is displayed

Visibility: public

Abstract: No

Method: Object Usage information is stored in the associated Relation Set database. The path to the corresponding Relation Set database is passed when creating the frame. Object Usage info is queried using SQLite API and SQL syntax.

##### **4.1.12.5.7 State Behavior**

No significant state behavior defined.

##### **4.1.12.5.8 L.O.S.**

See 4.1.7.2.5

##### **4.1.12.5.9 Constraints**

No constraints defined.

#### **4.1.12.6 MarketBasketRelationSet**

##### **4.1.12.6.1 Purpose**

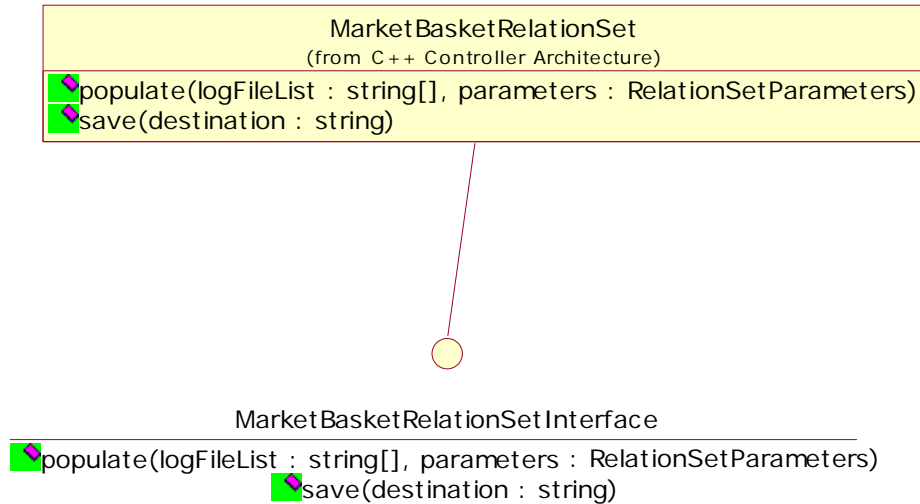
Purpose of this class is to provide functionality that allows to generate relation sets based on Log Files and to store the resulting relations in database format.

##### **4.1.12.6.2 Defined In: SC-03**

C++ Controller

**4.1.12.6.3 Interface**

See 4.1.7.3.2.1, 4.1.7.3.2.2

**4.1.12.6.4 Parameters**

No parameters defined

**4.1.12.6.5 Attributes**

No attributed defined

**4.1.12.6.6 Operations****4.1.12.6.6.1 Populate (OP-11)**

Identifier: OP-11

Operation name: Populate

Parameters: List of paths to source Log Files. start date, end date

Result: None

Purpose: Allows to create set of relations based on input Log Files and parameters

Pre-conditions: Instance of MarketbasketRelationSet is created. Source Log Files are available on local machine

Post-conditions: Internal representation of RelationSet is created according to the algorithm and based on the provided input data

Visibility: public

Abstract: No

Method: See 4.1.7.3.2.1

#### **4.1.12.6.6.2 Save (OP-11)**

Identifier: OP-11

Operation name: Save

Parameters: Path to save RelationSet database to

Result: None

Purpose: Allows to store internal representation of RelationSet in form of database that could be queried by other components

Pre-conditions: Instance of RelationSet was created. Operation populate was called.

Post-conditions: SQLite database is created at the specified path and contains representation of the previously populated RelationSet

Visibility: public

Abstract: No

Method: Uses SQLite API and tools to create and populate a database file.

#### **4.1.12.6.7 State Behavior**

No significant state behaviors defined

#### **4.1.12.6.8 L.O.S.**

See 4.1.7.3.5

#### **4.1.12.6.9 Constraints**

No constraints defined

### **4.1.12.7 ClusteringObjectTree**

#### **4.1.12.7.1 Purpose**

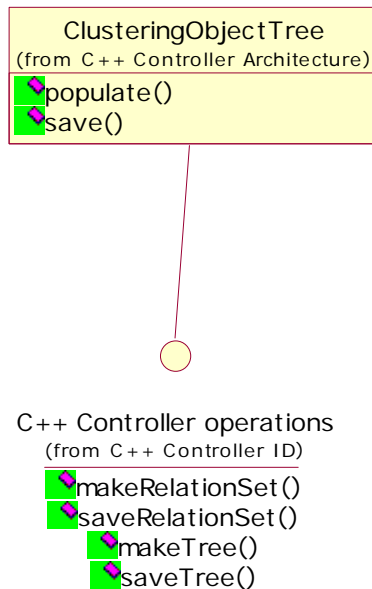
Purpose of this class is to provide functionality allowing creation of ObjectTrees based on source RelationSet and storing ObjectTree in format that is supported by the H3Viewer component.

#### **4.1.12.7.2 Defined In: SC-03**

C++ Controller

### 4.1.12.7.3 Interface

See 4.1.7.3.2.3, 4.1.7.3.2.4



**Figure 70 Clustering Object Tree Interface**

### 4.1.12.7.4 Parameters

No parameters defined

### 4.1.12.7.5 Attributes

No attributes defined

### 4.1.12.7.6 Operations

#### 4.1.12.7.6.1 Populate (OP-12)

Identifier: OP-12

Operation name: populate

Parameters: Path to source RelationSet database. Granularity of clustering

Result: None

Purpose: Allows to generate an ObjectTree based on source RelationSet using centrality analysis and graph clustering algorithm.

Pre-conditions: The specified source RelationSet was properly generated.

Post-conditions: Internal representation of ObjectTree is populated.

Visibility: public

Abstract: No

Method: See 4.1.7.3.2.3

#### **4.1.12.7.6.2 Save (OP-13)**

Identifier: OP-13

Operation name: save

Parameters: Path to save Object Tree.

Result: None

Purpose: Allows to save internal representation of Object Tree in format that is supported by the H3Viewer component.

Pre-conditions: Internal representation of Object Tree was properly populated.

Post-conditions: Object Tree is saved at the specified location

Visibility: public

Abstract: No

Method: 4.1.7.3.2.4

#### **4.1.12.7.7 State Behavior**

No significant behavior states defined

#### **4.1.12.7.8 L.O.S.**

See 4.1.7.3.5

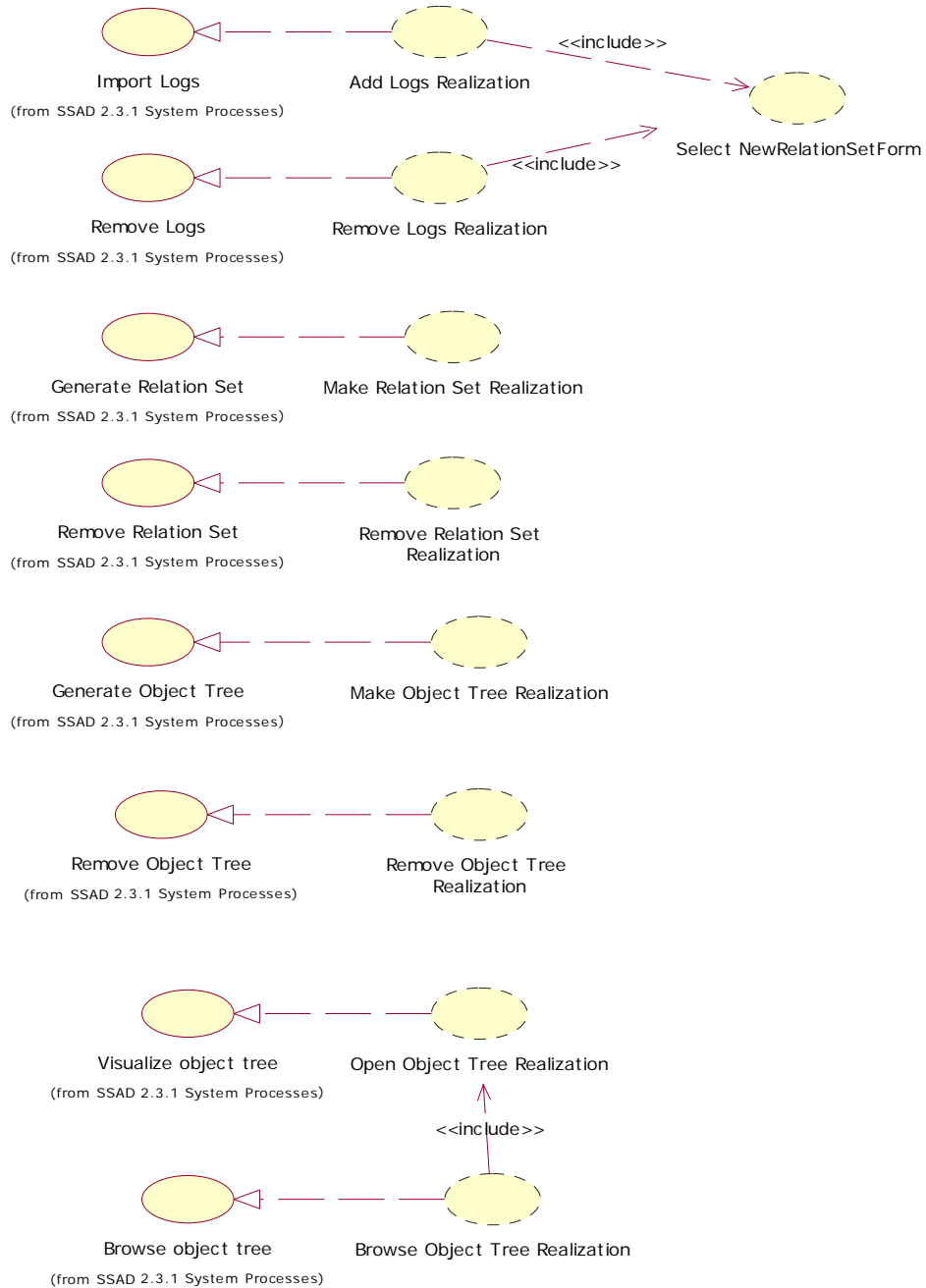
#### **4.1.12.7.9 Constraints**

No constraints defined.

### **4.1.13 Objects**

No objects defined.

## 4.2 Behavior



**Figure 71 System Processes Realization**

### 4.2.1 Add Logs Realization

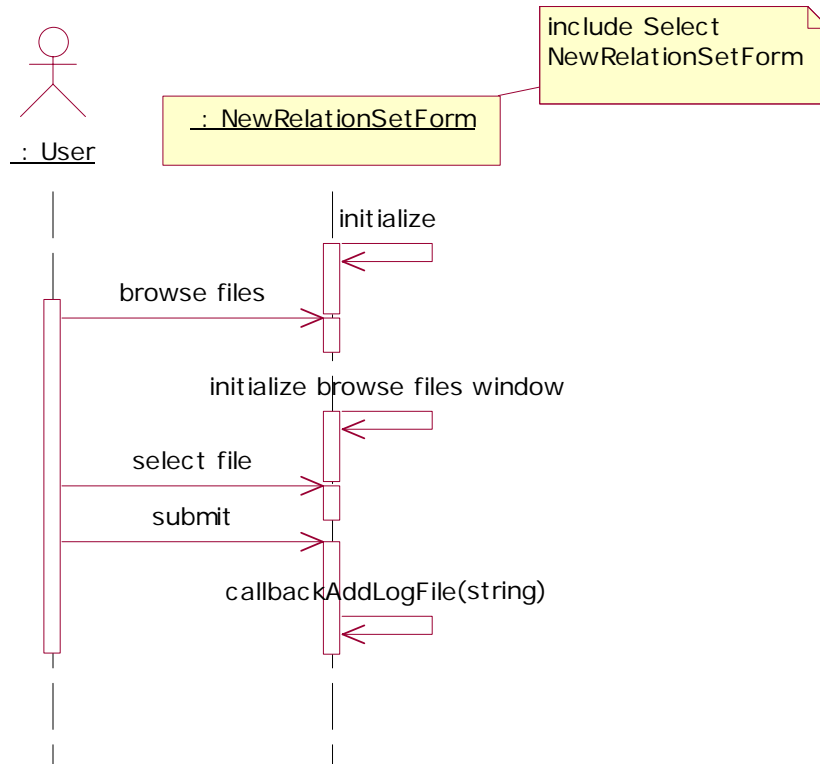


Figure 72 Add Logs Realization Sequence Diagram



### 4.2.3 Make Relation Set Realization

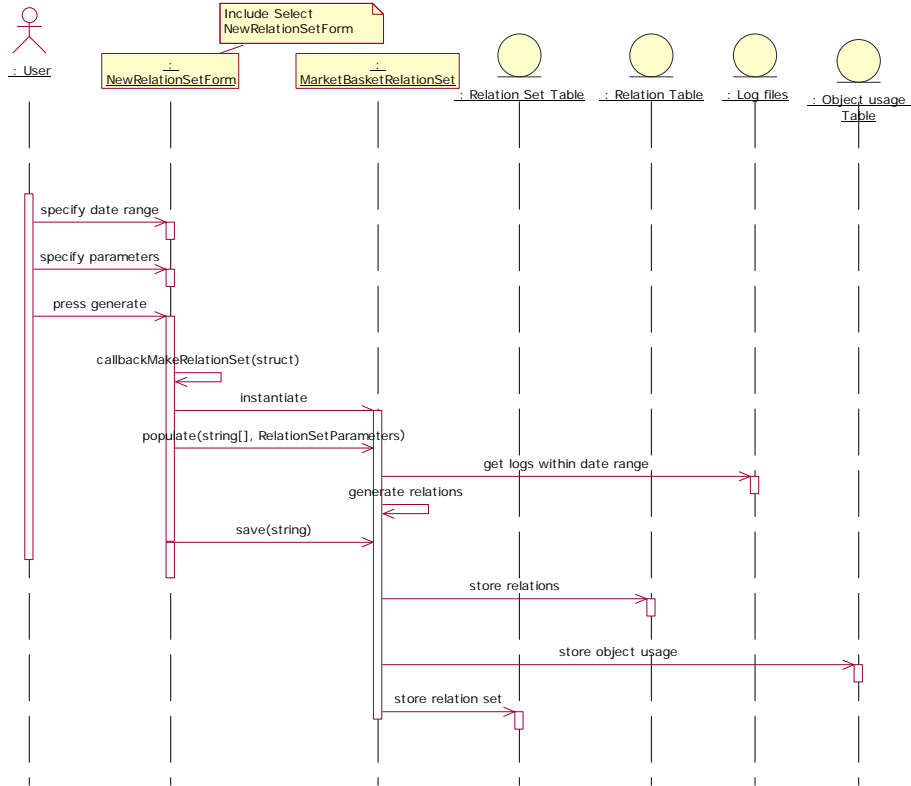


Figure 74 Make Relation Set Realization

## 4.2.4 Remove Relation Set Realization

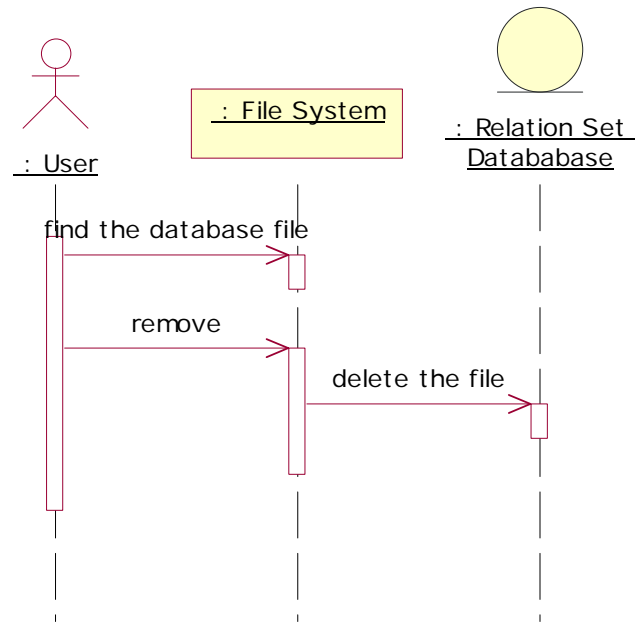


Figure 75 Remove Relation Set Realization

### 4.2.5 Make Object Tree Realization

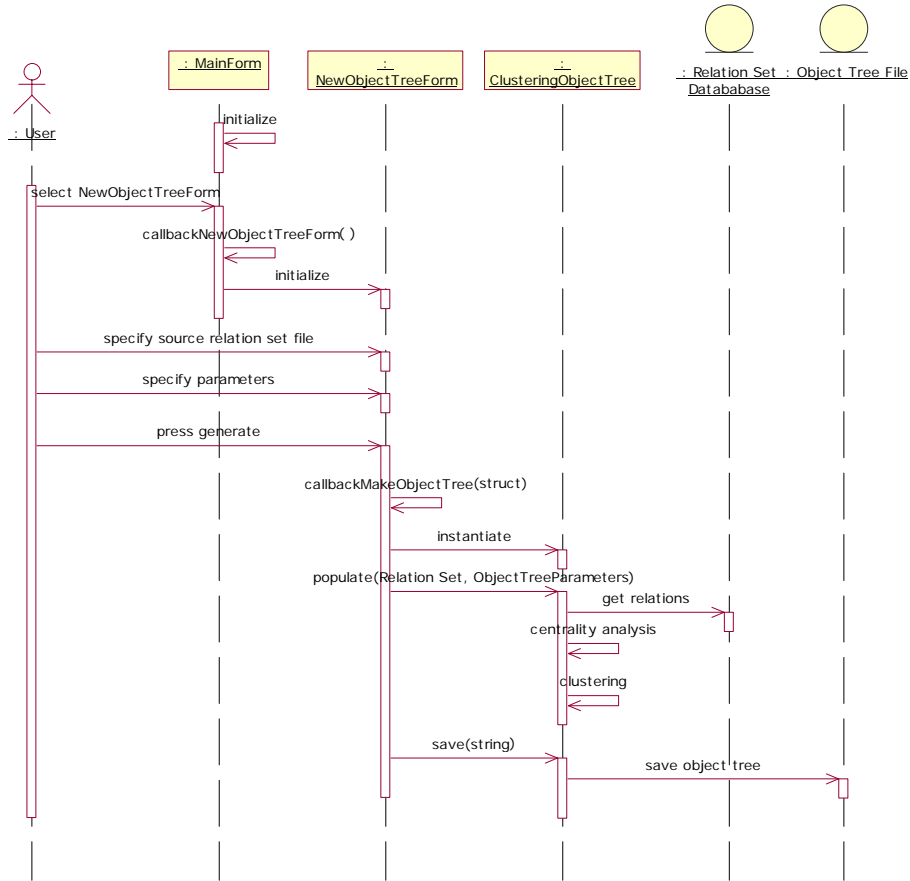


Figure 76 Make Object Tree Realization

## 4.2.6 Remove Object Tree Realization

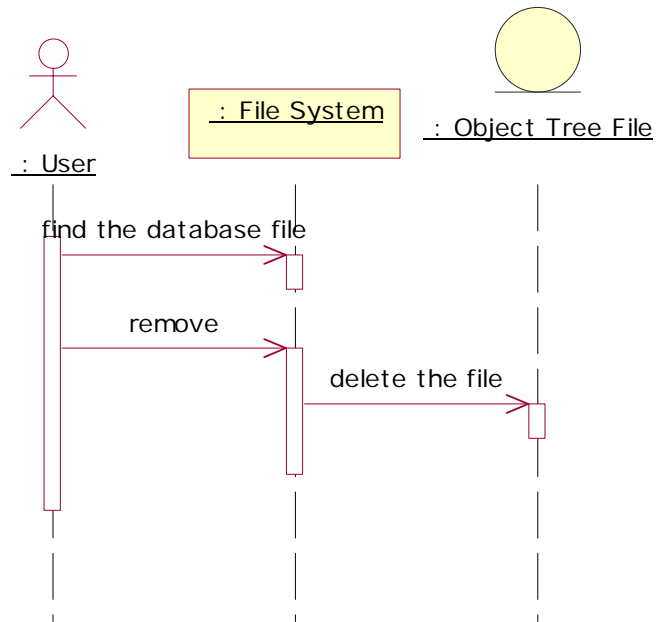


Figure 77 Remove Object Tree Realization

### 4.2.7 Open Object Tree Realization

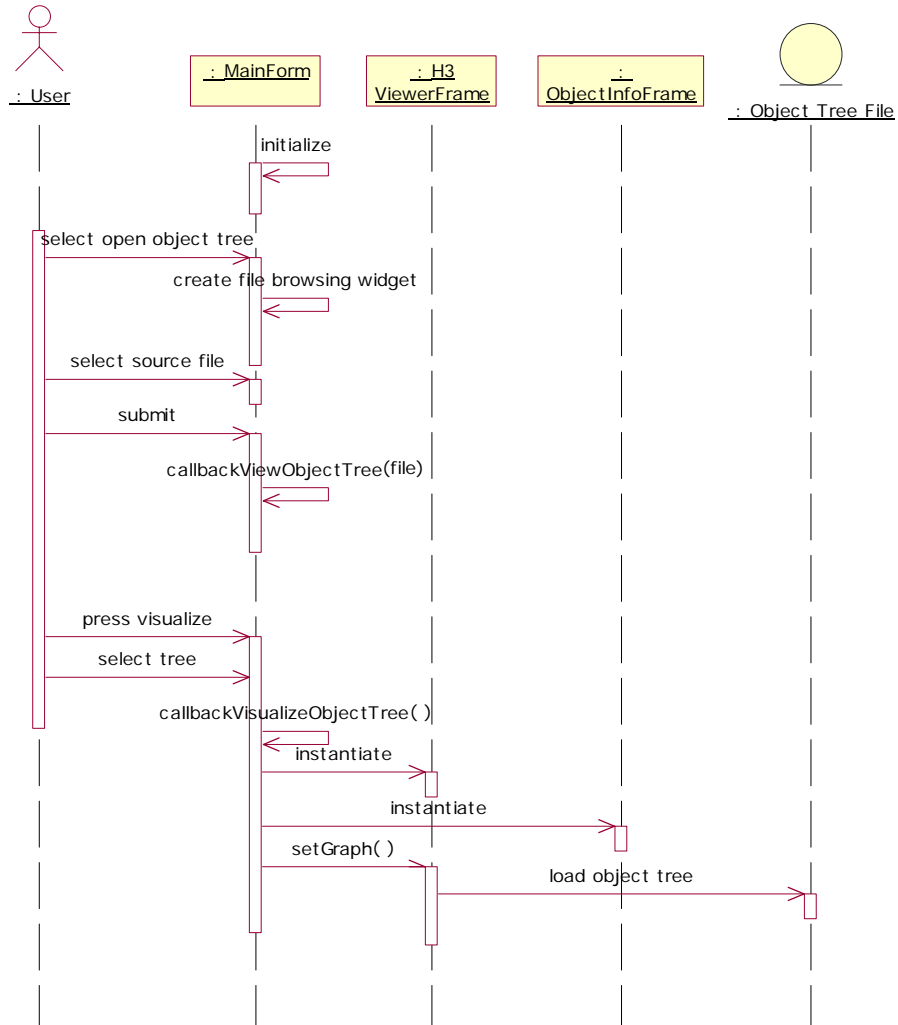


Figure 78 Open Object Tree Realization

### 4.2.8 Browse Object Tree Realization

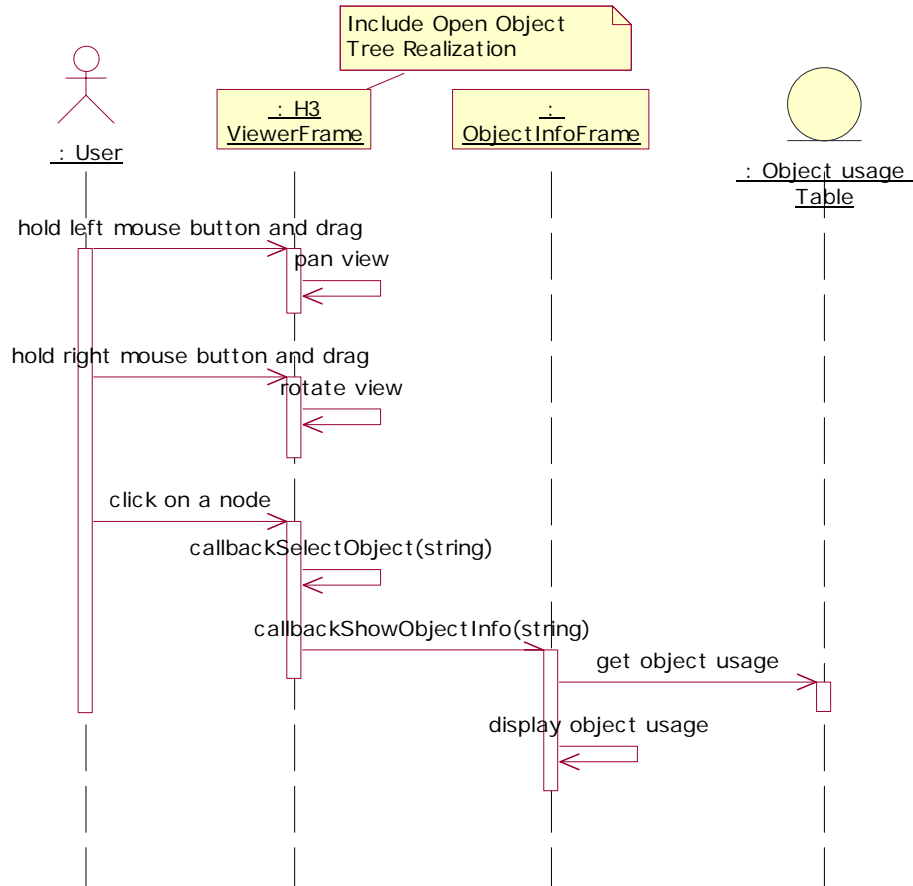


Figure 79 Browse Object Tree Realization

### 4.3 L.O.S. projected

L.O.S Goal	Applies To	How	Projected Value	Evaluation Technique
LR-1 [SSRD 5]: Dependability	SCO-01, SCO -02, SCO -03, SCO -04	Equally	Pass 95% of test cases	Estimation
LR-2 [SSRD 5]: Usability	SCO -04	Equally	Satisfied with the client	Estimation
LR-3 [SSRD 5]:	SCO -03	Equally	Pass 95% of	Estimation

Operability in multitasking environment			test cases	
LR-4[SSRD 5]: Performance on data of current scale	SCO -04	Equally	Able to perform properly with 10% increase.	Estimation

Table 16 L.O.S projected.

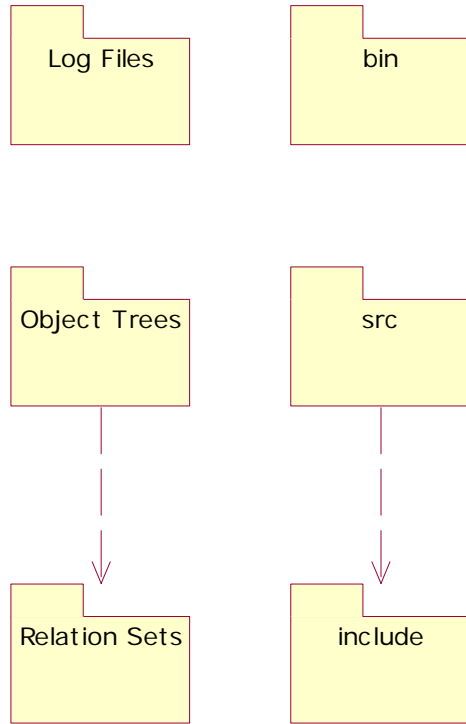
## 4.4 Patterns & Frameworks

We have chosen multi-layer pattern architecture.

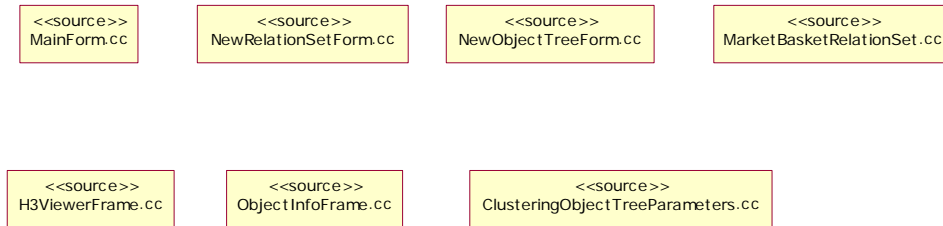
Name	Description	Benefits, Cots & limitations
Multi-layer Pattern	Multi-Layer system is a kind of software system and uses multi-layer architecture pattern. See <a href="http://www.site.uottawa.ca:4321/oose/index.html#multi-layer_table">http://www.site.uottawa.ca:4321/oose/index.html#multi-layer_table</a>	It replaces a layer by an improved version, or by one of different capabilities. In addition, it also increases reusability. It divides and conquers since the separate layers can be independently designed.

Table 17 Architectural Styles, Patterns &amp; Frameworks

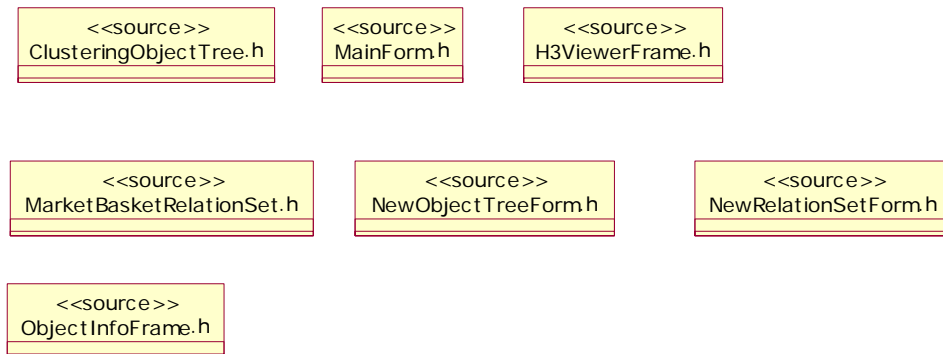
## 4.5 Project Artifacts



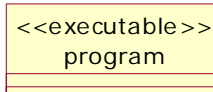
**Figure 80 Directory structure for Project**



**Figure 81 Src directory content**



**Figure 82 Include directory content**



```
<<executable>>
program
```

Figure 83 Bin directory content

## 5. Glossary for System Analysis and Design

**DA- Digital Archive :** Collection of electronic and multi-media documents, such as images, photos, audios, and videos

**GUI:** graphical user interface.

**Log Data:** they are the data gathered when users visit the library's digital achieves, they contain: user's IP address, item's id, and the time of visit.

**LANL :** Los Alamos National Laboratory.

**L.O.S –** Level of Service

**MBASE:** Model-based System Architecting and Software Engineering (MBASE).

**UML:** Unified Modeling Language (UML).

## 6. Appendices

N/A