Publication of Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th April 2011. Vol. 26 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SINGLE REPOSITORY FOR SOFTWARE COMPONENT SELECTION (SRSCS): A REUSABLE SOFTWARE COMPONENT SELECTION TECHNIQUE

[1]YOUNAS WAHAB, [2]MUHAMMAD IMRAN BABAR, [3]SHAHBAZ AHMED
[1]Research Scholar Department of Software Engineering, IIU, Islamabad, Pakistan
[2]Research Scholar, Department of Software Engineering, IIU, Islamabad, Pakistan
[3]Assistant Prof., Department of Software Engineering, IIU, Islamabad, Pakistan

ABSTRACT

This work contributes a novel approach for selection of reusable components based on Functional Requirements using SRSCS (Single Repository for Software Component Selection) mechanism. Different repositories are available for reusable components and to select the best qualifying component from those repositories, customers must visit all those one by one and will select the best qualifying component based on their requirements. SRSCS selects the reusable component based on the Functional Requirements, and as it is a black box process so the source, based on which the component is qualified, is the description of the component provided by the *Owner Organization.* SRSCS's main purpose is to provide a single point of the access to the customers from where they can select their required component instead of visiting all the repositories one by one. SRSCS extract information about the reusable component from different repositories, transform the description provided by the owner organization in order to classify information and remove the redundant information and store it on the single place i.e. SRSCS repository. SRSCS process consists of Extraction, Transformation, Loading and Component selection steps. In "*Extraction" step* information is extracted from the Owner Organization. In *Transformation step,* information is transformed, redundancy is removed and components are classified based on Type of the components. In *Loading step* the extracted information is loaded into the single SRSCS repository, and in the *Component Selection step*, customer sets the priorities of all information provided and the component is selected totally on the customer's choice. SRSCS has made the selection easy with improved time efficiency as compared to other approaches. All the components are available in one place and the results are accurate.

Keywords: *Extraction, Transformation, Loading, Component Selection Process, MultiLoad, FastExport*

1. INTRODUCTION

Presently data digitizing in all its possible forms is resulting in the need of best quality software systems in least possible time. A big challenge, that organizations are facing, is to provide a best quality product within least time. If they do so by hiring more developer and experts for development of the high quality product in less time then it results in high costs. Organizations are in the need to develop a high quality software product in less time domain and low budget. Component Based Software Engineering (CBSE) provides a solution to it. In CBSE, software is developed from built in available software components [7]. The available software components, tested in the same environment, are reusable. Through reusability a high quality and reliable product can be developed in less time and low cost [9]. Based on reusability software engineering is divided into Domain Engineering and Application Engineering [11]. In domain engineering, component for reuse, is created and stored in reuse library [12]. During Application Engineering required component is selected from the reuse library and is used according to the needs or requirements. The said research is focusing on the issue, that CBSE faces, is how to select the best qualifying component from available repositories. Qualification means to check how much a reusable component is according to our requirements [11]. Component selection is based on decision making which is complex in terms of initiation process i.e. from where to start for best results. Different authors have presented different approaches for the selection of reusable components. These approaches have certain limitations that are discussed in section 3.

Publication of Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th April 2011. Vol. 26 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In this paper SRSCS (Single Repository for Software Component Selection) approach is presented for component selection. SRSCS's main focus is to develop a central single repository from where Customers can select their required component through functional requirements.

Teradata is chosen for implementation because of the reason that we are extracting information from heterogeneous source e.g. flat file, relational databases etc and then after transformation we load that data into a single repository. SRSCS approach is based on ETL (*Extraction, Transformation, and Loading*) and Teradata provides the utilities like *FastLoad*, *MultiLoad* and *FastExport*. *Fastload* and *MultiLoad* help in loading data into a warehouse [17] from heterogeneous sources and through *FastExport* data is exported from Teradata RDBMS to Flat Files or Tables etc [18].

2. LITERATURE SURVEY

Component Based Software Engineering is used to develop high quality and reliable products in less time domain and low cost. Component Selection is one of the big challenges in CBSE and different authors have presented different techniques but lacking in desired performance, accuracy and user friendliness.

Maxym et.al. presented a "Semantic Component Selection Technique (SemaCS)". In SemaCS component's description is used to select and classify the reusable component, but there is a problem of huge amount of component's description and redundant information. It is not clearly defined that how the user will select the best required component from available components [10].

Vijayan et.al. presented "A Semantic-Based Approach to Component Retrieval" for reusable components retrieval [15]. The approach is based on a reuse repository where the components have been stored and a natural language interface has been used to interact with the repository in order to select the required components. The presented approach is based on the following steps.

Initial Query Generation
Query Refinement
Component Retrieval and feedback

User will write query in natural language and a natural language processing will be applied to that query to transform it to a structured query language. After transforming the user's query to structured query language, Query Refinement step is applied on it to check that query has been written correctly or not and lastly the reusable component is provided to the User.

Haining et.al. presented, *"Towards A Semantic-based Approach for Software Reusable Component Classification and Retrieval", an* approach for classification and retrieval of the software components [6]. In this approach the user communicates with the reuse repository in natural language. The component has been retrieved by semantic matching Components Semantic description and user query semantic representation against the Domain Ontology.

The user enters the requirements in an unrestricted natural language. An intelligent natural language interface transforms this query into conceptual graph semantic representation within a knowledge base and is translated into semantic web based representation. The software component functionality is identified by an analysis and annotation tool and is translated into semantic web based representation. Finally semantic matchmaker compares the component description and the user query in the conceptual graph and retrieves the required component.

Jeffrey et.al. presented "Melding Structured Abstracts anti the World Wide Web for Retrieval of Reusable Components" for retrieval of reusable components [8]. In this approach World Wide Web browser "mosaic" is used for Reusable Software Libraries (RSL). It showed the way to access the component quickly and to submit a component to RSL using the "Structured Abstract" of reusable components.

Young et.al. presented "Retrieving Software Components by Execution" approach for retrieval of reusable component from reuse library [13]. The approach is based on execution of the components by providing inputs generated systematically and based on these inputs it is decided either the component is needed or not.

A. Mili et.al. presented a refinement based system where the repository is based on formal representation and a binary relation is used to present program specification [1]. But the technique is based on assumptions and requires Formal Method skills to be followed practically. Bernd Fischer presented a specifications based retrieval for the selection of the reusable component [2]. Yonghao et.al. presented a way that how to Formalize and automate software reuse by using generality relation [16]. Gerald et.al. presented an automated approach for supporting software reuse through reverse engineering [5]. Most of the techniques are based on assumptions and there is the lack of user involvement. Vijayan et. al. have given some different touch to component selection and they used domain model and object libraries to identify software components [14]. The Domain Analyst performs analysis and builds a domain model. But in the same domain different components for the same purpose are developed by different organization. To select the best component the user must know about the functionalities provided by the components, but unfortunately, this method does not provide any functional information for decision making about component selection. The keywords based approach is presented in it. The keyword is retrieved from the user's query written in natural language. The keywords are mapped against the repository and the required components are selected.

As discussed above all the approaches have certain limitations and all organizations have their own development standards [3]. The above approaches do not provide the solution that how to select the best qualifying component among the available different repositories and how to compare components with each other if there are more than one component developed for the same purpose by different organization. The main focus of the SRSCS approach proposed in this thesis is to make a single repository and to transform the components that have been developed for same purpose into just one standard i.e. Customer's standard. Instead of visiting all the available repositories individually, SRSCS makes the Customers able to select their component from common SRSCS repository and makes it possible to compare components developed for the same purpose.

3. THE PROBLEM

In Component Based Software Engineering there are two types of organizations i.e. Owner Organization, who will be responsible for making reusable component, and the Customer who will select the reusable component and will reuse it.

Different Owner Organizations are busy in making the reusable components. Every Owner Organization has its own development standard [3] in order to develop the reusable components and to store it in the library by describing component's functionality. Different components for the same purpose can be developed by different organizations and every organization describes their component's description in different way. Customers face the problems in selection as the components are described in different way and are stored in different locations. The questions which may come in the mind of a customer are:

Q1. How many components are available for this purpose?

Q2. If more than one component available for this functionality then should I check all of them individually?

Q3. How will I compare all available components to select the one that fully qualifies my functional requirements?

Q4. Is there a simple way to select one best-required component in just a single go?

SRSCS's main purpose is to make a single repository, and to transform the components functionalities for the purpose to remove redundant information and to classify it in order to store it in a single repository. SRSCS will reduce the customers' frequent visits to different repositories.

4. SRSCS PROCESS

The main purpose of SRSCS is to provide a single point of access from where Customers can select their required component based on Functional requirements. Different reusable component are available in different repositories developed by different organizations. In SRSCS information is extracted about the reusable components developed by different Owner Organization and is brought into one standard way of description by performing transformation on it and then it is loaded into SRSCS repository. As shown in Fig. 1, SRSCS process consists of the following four steps:

- ➢ Extraction
- ➢ Transformation
- ➢ Loading
- ➢ Component Selection Process

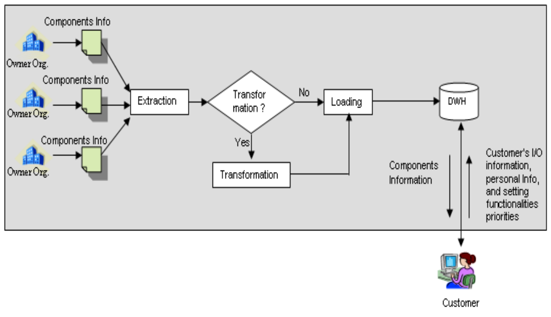


Fig. 1 SRSCS Process

Publication of Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th April 2011. Vol. 26 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In SRSCS process the information about the reusable components is coming from heterogeneous sources like Flat files, Excel Sheet, RDBMS tables etc. In SRSCS the main source of information in order to select a reusable component is the description of the component which is provided by the owner organization. There is no standard way of description of the reusable components [10], so the description provided by different organizations is not a standard one.

Further there can be more than one components, for the same purpose, that can be stored in different locations so in order to select the one best component the customer will face the problem as explained.

SRSCS brings information from different repositories to a single repository. The problem is that how to store information of different formats and how to select the best-required component quickly. In order to collect all the information from different available repositories, SRSCS performs *Extraction step*, which is the first step of the SRSCS process and gets all the required information about the reusable components. After *Extraction Transformation* is performed on it. *Transformation* is used to bring the description in one format and also to classify the components on the basis of Type of the component.

Then the extracted information is loaded into a single repository. The last step in SRSCS's process is *Component selection process*, in this stage the SRSCS's repository has all the required information and the Customer can select the required component based on functional requirements just by changing the priorities of the Functionalities and IO information. The component's qualification is determined by the "Component Qualification Equation" i.e. Eq.1.

4.1 Detailed SRSCS Process

The detailed SRSCS process is shown in Fig.2. Some technical details of the SRSCS process are discussed in this sections that how the Common Reusable Repository is built and how the data is stored in the repository of SRSCS and how reusable component is provided to the *Customer* from the repository.

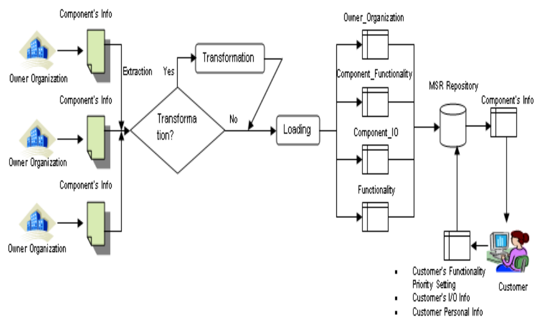


Fig. 2 SRSCS Detailed Process

4.1.1 Extraction

For making the repository, the important information about reusable components is extracted in order to transform and load it into the repository. For making the reuse library, the following information is required from the Owner Organization about the component.

- *Information about the Owner Organization*
  - Owner Organization name
  - Contact Detail
  - Component they are providing

- *Information about the Component's Functionality*
  - Functionality provided by the component
  - Purpose

- *Information abut Component's Input Output*
  - IO description of all functionalities

4.1.2 Transformation

Based on available information the reuse library is checked for available component for the same purpose. If a component is available then the transformation is applied on it based on its functionality in order to classify it by its type. If there is no such component in the reuse library then transformation is applied in order to bring it to a standard way of description without keeping any stored component in mind.

4.1.3 Loading

After *Extraction* and *Transformation* the next step is *Loading*. In *Loading, all* the Extracted and Transformed information is loaded into the

Publication of Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th April 2011. Vol. 26 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

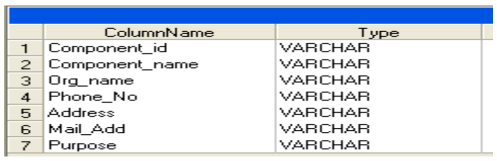
repository in order to make it available for the *Customers*. In Loading following steps are performed.

4.1.3.1 Loading Information About Owner Organization

After *Extraction* and *Transformation* the information is loaded into *Owner Organization Table*.

Component_id: *To uniquely identify the component*
Component_name: *name of the reusable component*
Org_name: *Owner Organization's name who has developed this component*
Phn_no: *Phone Number of the Owner Organization*
Address : *Address of the Owner Organization*
Mail_add: *Mail address of the Owner Organization*
Purpose: *Specifies the purpose of the developed component*

Figure 3 shows the Owner Organization Table in the CBSE database.

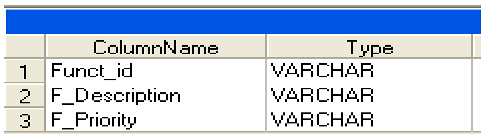


	ColumnName	Type
1	Component_id	VARCHAR
2	Component_name	VARCHAR
3	Org_name	VARCHAR
4	Phone_No	VARCHAR
5	Address	VARCHAR
6	Mail_Add	VARCHAR
7	Purpose	VARCHAR

Fig. 3 Owner Organization Table

4.1.3.2 Loading Information About Component's Functionality

In Functionality Table of SRSCS database the information about components' functionality is stored. Functionality Table keeps the following information about the reusable components as shown in Fig.4.



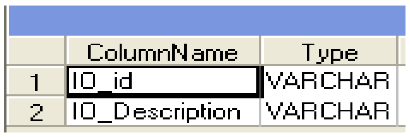
	ColumnName	Type
1	Funct_id	VARCHAR
2	F_Description	VARCHAR
3	F_Priority	VARCHAR

Fig.4 Functionality Table

Funct_id: *To uniquely identify functionality.*
F_Description: *Description about the functionality of the component.*
F_Priority: *Assign priority to each functionality*

4.1.3.3 Component's Functionality Result (I/O) Information

To select the best required component on the basis of functional requirements, information about the Input and Output of all the functionalities that it performs is also required. In SRSCS process the information about IO is extracted and is stored it in the repository as shown below in Fig. 5.



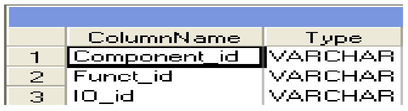
	ColumnName	Type
1	IO_id	VARCHAR
2	IO_Description	VARCHAR

Fig. 5 Component_IO Table

4.1.3.4 Relating Component's Functionality and Result (I/O) Information

SRSCS repository provides information about the reusable components to Customers for easy selection of the required component. SRSCS keeps the information about the Component's Functionality and I/O of that functionality as shown in Figure.6

Component_id: *To uniquely identify the component*
Funct_id: *To uniquely identify functionalities.*
IO_Id: *To uniquely identify IO information*



	ColumnName	Type
1	Component_id	VARCHAR
2	Funct_id	VARCHAR
3	IO_id	VARCHAR

Fig. 6 Component_IO table

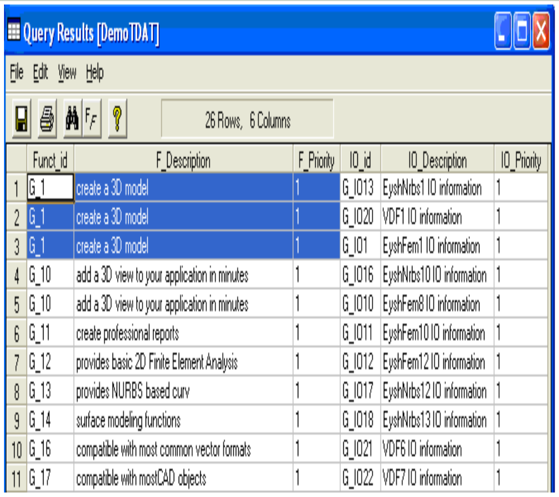
4.1.3.5 Component's Selection Process

Component selection process is mainly consists of the following steps.

i) *Information about the Customer*
ii) *Changing Functionality and I/O Priority Values*

SRSCS provides detailed information about the components to make the Customers able to select the best required component based on requirements. SRSCS provides information about the Component's Functionality and IO information of all functionalities that component performs. Fig.7 shows the information that SRSCS will provide to the Customers for Component selection.

Publication of Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th April 2011. Vol. 26 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



	Funct_id	F_Description	F_Priority	IO_id	IO_Description	IO_Priority
1	G_1	create a 3D model	1	G_IO13	EyshNrbs1 IO information	1
2	G_1	create a 3D model	1	G_IO20	VDF1 IO information	1
3	G_1	create a 3D model	1	G_IO1	EyshFem1 IO information	1
4	G_10	add a 3D view to your application in minutes	1	G_IO16	EyshNrbs10 IO information	1
5	G_10	add a 3D view to your application in minutes	1	G_IO10	EyshFem8 IO information	1
6	G_11	create professional reports	1	G_IO11	EyshFem10 IO information	1
7	G_12	provides basic 2D Finite Element Analysis	1	G_IO12	EyshFem12 IO information	1
8	G_13	provides NURBS based curv	1	G_IO17	EyshNrbs12 IO information	1
9	G_14	surface modeling functions	1	G_IO18	EyshNrbs13 IO information	1
10	G_16	compatible with most common vector formats	1	G_IO21	VDF6 IO information	1
11	G_17	compatible with mostCAD objects	1	G_IO22	VDF7 IO information	1

Fig. 7 SRSCS Customer's Information Table

4.1.3.6 Information about Customer

While selecting the component, Customer provides the following personal information.

Cust_id *Uniquely identifies the Customer*
Cust_nam *Shows the name of the Customer*
Cust_phone_number *Shows the Phone number of the Customer*
Cust_Address *Shows the address of the Customer*
Cust_Mail_Address *Shows the mail address of the Customer*

4.1.3.7 Changing Functionality and I/O Priority Values

As shown in the SRSCS process after loading all the information about the reusable components, the information will be available in a single location, and the customers can access and select the required component from that single location. As shown in Fig. 7 the component's functionalities and IO information have the priorities based on which Customer selects the required component.

Component Selection process is mainly based on the priorities of the Functionalities and its associated IO information, which the Customer will assign during the Component Selection Process. Functionality and IO can have any one of the following Priority Values.

H:	High priority
M:	Medium Priority
L:	Low Priority
N:	Not Required

For calculation purposes priority values are assigned which the Customer will assign during the Component's selection process.

H= 0.6……………….1.0
M= 0.3……………….0.5
L= 0.1……………….0.2
N=0

The following formula will be used to determine the component's qualification according to Customer's requirements.

$$\mathbf{Q_c = \sum_{i=1\ldots N} (FPi * IOi) / T * 100\ldots\ldots (eq.1)}$$

Where **FP** is Functionality Priority, **N** is the number of functionalities offered by a given component, **T** is Total Number of unique transformed Functionalities for the same purpose and $\mathbf{Q_c}$ shows percent qualification of component.

By default in Fig. 7 all the Functionalities and their associated IO information have H priority value i.e. 1. Customer can change this value at time of component selection for best results.

5. VALIDATION

In order to validate SRSCS a best graphic component will be selected, according to SRSCS criterion, among the available four graphic components for the same purpose.

Eyeshot Fem

create a 3D model|
change a 3D model|
can be used with models imported from other programs|
can perform professional shading|
can perform professional projection|
can perform professional zoom|
can perform professional pan|
can perform professional rotate|
can perform professional selection|
add a 3D view to your application in minutes|
create professional reports|
provides basic 2D Finite Element Analysis|

Eyeshot Nurbs

create a 3D model|
change a 3D model|
can be used with models imported from other programs|

Publication of Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th April 2011. Vol. 26 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

can perform professional shading|
can perform professional projection|
can perform professional zoom|
can perform professional pan|
can perform professional rotate|
can perform professional selection|
add a 3D view to your application in minutes|
create professional reports|
provides NURBS based curve|
surface modeling functions

Eyeshot

create a 3D model|
change a 3D model|
can be used with models imported from other programs|
can perform professional shading|
can perform professional projection|
can perform professional zoom|
can perform professional pan|
can perform professional rotate|
can perform professional selection|
add a 3D view to your application in minutes|
create professional reports|
provides polygon based 3D modeling functions|

VectorDraw Developer Framework (VDF)

create 2D drawings|
create 3D drawings|
manage 2D drawings|
manage 3D drawings|
print 2D and 3D drawings|
compatible with most common vector formats|
compatible with mostCAD objects|
supports over 10 vector formats|
supports many raster formats|
fully object oriented|

To select the best qualifying component among the available four graphics based on Customer choice, the SRSCS approach is implemented step by step.

5.1.1 Extraction

During *Extraction* the following information about the components will be extracted in SRSCS process.

Component's Name
Owner Organization Name
Phone #
Address
Mail Add
Purpose
Component's Functionalities

In this case there are four graphic components, so the information will be extracted about all the four as follows.

Component No. 1

Component's Name:	Eyeshot Fem
Owner Organization Name:	USoft
Phone #:	+92-51-9332624
Address:	Street No. 152, H#59, G-9/4 Islamabad Pakistan
Mail Add:	Info@usof.com
Purpose:	Graphics

Component's Functionalities:

F#	Functionalities	Input and Output Information
EyshFem1	create a 3D model|	EyshFem1 IO information
EyshFem2	change a 3D model|	EyshFem2 IO information
EyshFem3	can be used with models imported from other programs|	EyshFem3 IO information
EyshFem4	can perform professional shading|	EyshFem4 IO information
EyshFem5	can perform professional projection|	EyshFem5 IO information
EyshFem6	can perform professional zoom|	EyshFem6 IO information
EyshFem7	can perform professional pan|	EyshFem7 IO information
EyshFem8	can perform professional rotate|	EyshFem8 IO information
EyshFem9	can perform professional selection|	EyshFem9 IO information
EyshFem10	add a 3D view to your application in minutes|	EyshFem10 IO information
EyshFem11	create professional reports|	EyshFem11 IO information
EyshFem12	provides basic 2D Finite Element Analysis|	EyshFem12 IO information

Component No. 2

Component's Name:	Eyeshot Nurbs
Owner Organization Name:	Inova
Phone #:	+92-51-9332655
Address:	Street No. 48, H#595, F-6/4 Islamabad Pakistan
Mail Add:	Info@inova.com
Purpose:	Graphics

Component's Functionalities:

F#	Functionalities	Input and Output Information
EyshNrbs1	Create a 3D model|	EyshNrbs1 IO information
EyshNrbs2	Change a 3D model|	EyshNrbs2 IO information
EyshNrbs3	can be used with models imported from other programs|	EyshNrbs3 IO information
EyshNrbs4	can perform professional shading|	EyshNrbs4 IO information
EyshNrbs5	can perform professional projection|	EyshNrbs5 IO information
EyshNrbs6	can perform professional zoom|	EyshNrbs6 IO information
EyshNrbs7	can perform professional pan|	EyshNrbs7 IO information
EyshNrbs8	can perform professional rotate|	EyshNrbs8 IO information
EyshNrbs9	can perform professional selection|	EyshNrbs9 IO information
EyshNrbs10	add a 3D view to your application in minutes|	EyshNrbs10 IO information
EyshNrbs11	Create professional reports|	EyshNrbs11 IO information
EyshNrbs12	provides NURBS based curv|	EyshNrbs12 IO information
EyshNrbs13	Surface modeling functions|	EyshNrbs13 IO information

Component No. 3

Component's Name:	Eyeshot
Owner Organization Name:	Techlogix
Phone #:	+92-51-9232655
Address:	Street No. 105, H#678, H-10/4 Islamabad Pakistan
Mail Add:	Info@techlogix.com
Purpose:	Graphics

Component's Functionalities:

F#	Functionalities	Input and Output Information
Eysh1	create a 3D model|	Eysh1 IO information
Eysh2	change a 3D model|	Eysh2 IO information
Eysh3	can be used with models imported from other programs|	Eysh3 IO information
Eysh4	can perform professional shading|	Eysh4 IO information
Eysh5	can perform professional projection|	Eysh5 IO information
Eysh6	Can perform professional zoom|	Eysh6 IO information
Eysh7	Can perform professional pan|	Eysh7 IO information
Eysh8	Can perform professional rotate|	Eysh8 IO information
Eysh9	Can perform professional selection|	Eysh9 IO information
Eysh10	Can perform professional selection|	Eysh10 IO information
Eysh11	Add a 3D view to your application in minutes|	Eysh11 IO information
Eysh12	create professional reports|	Eysh12 IO information
Eysh13	Provides polygon based 3D modeling functions|	Eysh13 IO information

Component No. 4

Component's Name:	VectorDraw Developer Framework (VDF)
Owner Organization Name:	ISoft
Phone #:	+92-51-9112655
Address:	Street No. 48, H#595, F-6/3 Islamabad Pakistan

Mail Add:	Info@isoft.com
Purpose:	Graphics

Component's Functionalities:

F#	Functionalities	Input and Output Information
VDF1	Create 2D drawings|	VDF1 IO information
VDF2	Create 3D drawings|	VDF2 IO information
VDF3	Manage 2D drawings|	VDF3 IO information
VDF4	Manage 3D drawings|	VDF4 IO information
VDF5	print 2D and 3D drawings|	VDF5 IO information
VDF6	compatible with most common vector formats|	VDF6 IO information
VDF7	compatible with mostCAD objects|	VDF7 IO information
VDF8	Supports over 10 vector formats|	VDF8 IO information
VDF9	Supports many raster formats|	VDF9 IO information
VDF10	fully object oriented|	VDF10 IO information

After Extraction, the required information about all four graphic components will be available.

The *Component's Functionalities* portion of every component consists of two colors text. The *Green* color shows common functionalities which may have different description but will perform the same functionality, and the *other* color shows the unique functionality of the component.

5.1.2 Transformation

The information is coming from different sources, i.e. form different *Owner Organizations*, about graphic components possibly with same functionalities and IO information but with slight difference. The very information or description about components is provided by the organizations in different standards. Transformation will transform those functionalities that perform the same functions, but described in different way, into one function in order to remove the redundant information and the same process will also be performed for IO information. After transformation each component's functionalities and IO are presented in two different colors as shown in Fig. 8.

5.1.2.1 Green Color Description

The Green color shows that the functionalities or IO information are common among graphic components i.e. these functionalities or IO information are also provided by other components.

5.1.2.2 Other Color Description

The colors other than Green show different functionalities or IO of a component.

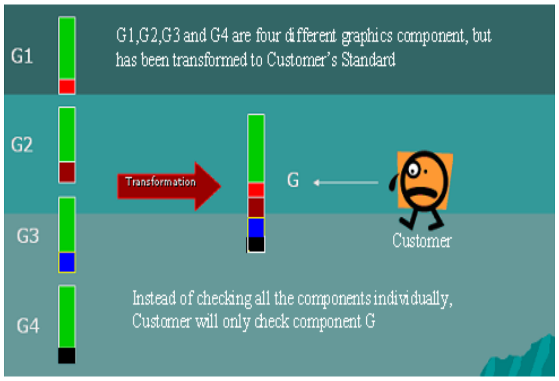

G1
G1,G2,G3 and G4 are four different graphics component, but has been transformed to Customer's Standard
G2
Transformation
G
G3
Customer
Instead of checking all the components individually, Customer will only check component G
G4

Fig. 8 Transformation

In Fig.8 the G1 represents Graphics Component No 1 i.e. *Eyeshot Fem*, G2 represents Graphics Component No 2 i.e. *Eyeshot Nurbs*, G3 represents Graphics Component No 3 i.e. *Eyeshot* and G4 represents Graphics Component No 4 i.e. *VectorDraw Developer Framework (VDF).*

After implementing transformation on G1, G2, G3 and G4, the component G is obtained with the following *Component Functionalities*.

Graphics Component G (after transformation)

F_id	F_Description
G_1	create a 3D model|
G_2	change a 3D model|
G_3	can be used with models imported from other programs|
G_4	can perform professional shading|
G_5	can perform professional projection|
G_6	can perform professional zoom|
G_7	can perform professional pan|
G_8	can perform professional rotate|
G_9	can perform professional selection|
G_10	add a 3D view to your application in minutes|
G_11	create professional reports|
G_12	Provides basic 2D Finite Element Analysis|
G_13	Provides NURBS based curv|

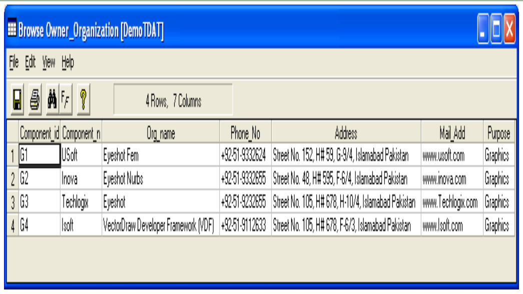
G_14	surface modeling functions|
G_15	Provides polygon based 3D modeling functions|
G_16	print 2D and 3D drawings|
G_17	compatible with most common vector formats|
G_18	compatible with most CAD objects|
G_19	Supports over 10 vector formats|
G_20	Supports many raster formats|
G_21	fully object oriented|

Component's IO:

IO_id	IO_Description
G_IO1	EyshFem1 IO information
G_IO2	EyshFem2 IO information
G_IO3	EyshFem3 IO information
G_IO4	EyshFem4 IO information
G_IO5	EyshFem5 IO information
G_IO6	EyshFem7 IO information
G_IO7	EyshFem9 IO information
G_IO8	EyshFem11 IO information
G_IO9	EyshFem6 IO information
G_IO10	EyshFem8 IO information
G_IO11	EyshFem10 IO information
G_IO12	EyshFem12 IO information
G_IO13	EyshNrbs1 IO information
G_IO14	EyshNrbs3 IO information
G_IO15	EyshNrbs6 IO information
G_IO16	EyshNrbs10 IO information
G_IO17	EyshNrbs12 IO information
G_IO18	EyshNrbs13 IO information
G_IO19	Eysh13 IO information
G_IO20	VDF1 IO information
G_IO21	VDF6 IO information
G_IO22	VDF7 IO information
G_IO23	VDF8 IO information
G_IO24	VDF9 IO information
G_IO25	VDF10 IO information

5.1.3 Loading

In SRSCS Process, after Extraction and Transformation the next step is to Load data into repository. The required information about component i.e. Owner Organization name, Contact Detail, Component name are loaded into Owner_Organization Table as shown in Fig. 9.

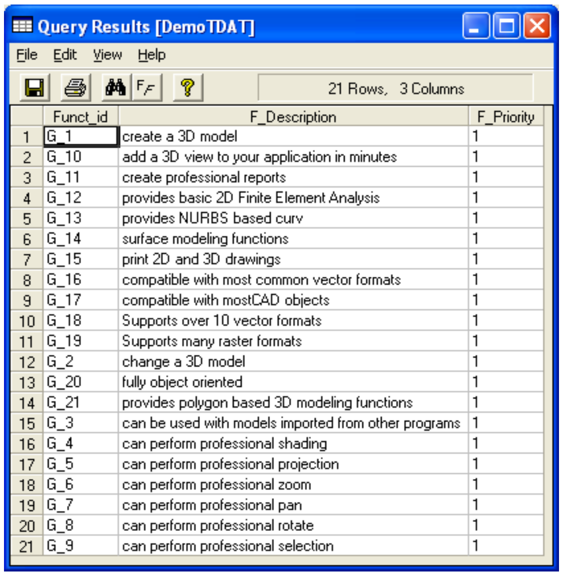


	Component_id	Component_n	Org_name	Phone_No	Address	Mail_Add	Purpose
1	G1	USoft	Eyeshot Fem	+92-51-9332624	Street No. 152, H# 59, G-9/4, Islamabad Pakistan	www.usoft.com	Graphics
2	G2	Inova	Eyeshot Nurbs	+92-51-9332655	Street No. 48, H# 595, F-6/4, Islamabad Pakistan	www.inova.com	Graphics
3	G3	Techlogix	Eyeshot	+92-51-9232655	Street No. 105, H# 678, H-10/4, Islamabad Pakistan	www.Techlogix.com	Graphics
4	G4	Isoft	VectorDraw Developer Framework (VDF)	+92-51-9112633	Street No. 105, H# 678, F-6/3, Islamabad Pakistan	www.Isoft.com	Graphics

Fig. 9 Owner_Organization Table

After loading information about Component's Owner organization, the information about component's functionalities and IO is loaded. The transformed functionalities and IO information i.e. of component G is loaded into Functionality and IO Tables as shown in Fig. 10 and Fig. 11.

The Functionality Table consists of the Funct_id, F_Description and Priority fields. The Priority field shows priority of functionality in the Table. During loading the default priority of all functionalities will be High i.e. 1.

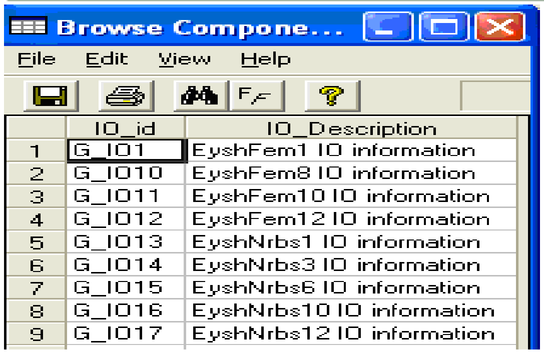


	Funct_id	F_Description	F_Priority
1	G_1	create a 3D model	1
2	G_10	add a 3D view to your application in minutes	1
3	G_11	create professional reports	1
4	G_12	provides basic 2D Finite Element Analysis	1
5	G_13	provides NURBS based curv	1
6	G_14	surface modeling functions	1
7	G_15	print 2D and 3D drawings	1
8	G_16	compatible with most common vector formats	1
9	G_17	compatible with mostCAD objects	1
10	G_18	Supports over 10 vector formats	1
11	G_19	Supports many raster formats	1
12	G_2	change a 3D model	1
13	G_20	fully object oriented	1
14	G_21	provides polygon based 3D modeling functions	1
15	G_3	can be used with models imported from other programs	1
16	G_4	can perform professional shading	1
17	G_5	can perform professional projection	1
18	G_6	can perform professional zoom	1
19	G_7	can perform professional pan	1
20	G_8	can perform professional rotate	1
21	G_9	can perform professional selection	1

Fig.10 Functionality Table

All the functionalities of components have Input and Output information. After extracting and transforming information about the functionalities IO, the IO information will be loaded into Component_IO table as shown in Fig. 11.

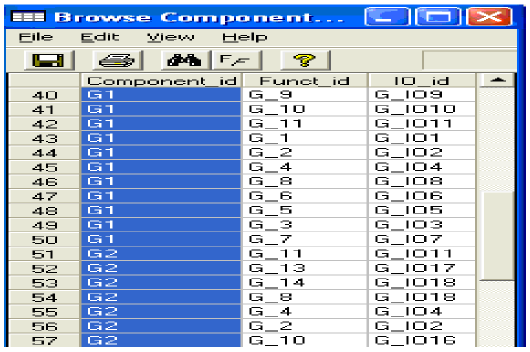
Publication of Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th April 2011. Vol. 26 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



	IO_id	IO_Description
1	G_IO1	EyshFem1 IO information
2	G_IO10	EyshFem8 IO information
3	G_IO11	EyshFem10 IO information
4	G_IO12	EyshFem12 IO information
5	G_IO13	EyshNrbs1 IO information
6	G_IO14	EyshNrbs3 IO information
7	G_IO15	EyshNrbs6 IO information
8	G_IO16	EyshNrbs10 IO information
9	G_IO17	EyshNrbs12 IO information

Fig. 11 Component_IO Table

In Owner_Organization Table, Functionality Table and Component_IO Table there is information about Organization, Functionalities and IO information of the all the components, but there is no way to find out that which functionality is performed by which component and which IO information belongs to which functionality. Table Component_Fucnt_IO as shown in Fig. 12 shows the relation between the component, functionality and IO information in order to determine that which functionality belongs to which component and have which IO information.



	Component_id	Funct_id	IO_id
40	G1	G_9	G_IO9
41	G1	G_10	G_IO10
42	G1	G_11	G_IO11
43	G1	G_1	G_IO1
44	G1	G_2	G_IO2
45	G1	G_4	G_IO4
46	G1	G_8	G_IO8
47	G1	G_6	G_IO6
48	G1	G_5	G_IO5
49	G1	G_3	G_IO3
50	G1	G_7	G_IO7
51	G2	G_11	G_IO11
52	G2	G_13	G_IO17
53	G2	G_14	G_IO18
54	G2	G_8	G_IO18
55	G2	G_4	G_IO4
56	G2	G_2	G_IO2
57	G2	G_10	G_IO16

Fig.12 Component_Funct_IO Table

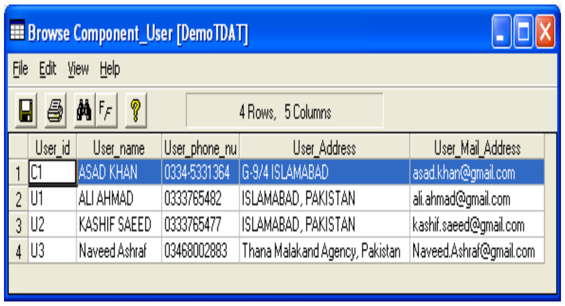
5.1.4 Component's Selection Process

In this step of the SRSCS process the Customer enters personal information and sets the priorities of the Functionalities and IO of the component.

5.1.5 Customer's Information

Customer will enter the personal information and will be stored in Customer Table as shown in Fig 13.

Cust_id: C1
Cust_name: ASAD KHAN
Cust_phone_number: 0334-5331364
Cust_Address: G-9/4 ISLAMABAD
Cust_Mail_Address: asad.khan@gmail.com

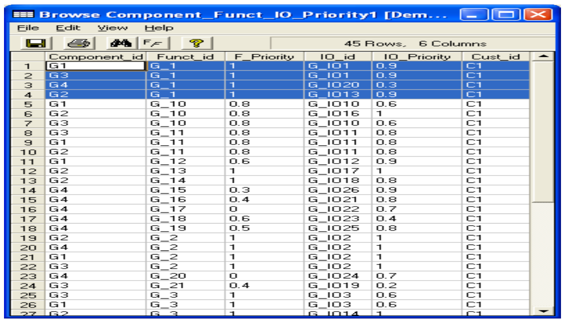


	User_id	User_name	User_phone_nu	User_Address	User_Mail_Address
1	C1	ASAD KHAN	0334-5331364	G-9/4 ISLAMABAD	asad.khan@gmail.com
2	U1	ALI AHMAD	0333765482	ISLAMABAD, PAKISTAN	ali.ahmad@gmail.com
3	U2	KASHIF SAEED	0333765477	ISLAMABAD, PAKISTAN	kashif.saeed@gmail.com
4	U3	Naveed Ashraf	03468002883	Thana Malakand Agency, Pakistan	Naveed.Ashraf@gmail.com

Fig. 13: Customer Table

5.1.5.1 Changing Priorities

Component selection is the last process of the SRSCS approach. After performing Extraction, Transformation and loading all the required information about the reusable component will be available in SRSCS repository. To select the best required component, the Customer will set the priorities of the two fields, i.e. F_Priority and IO_Priority. F_Priority will show that how much that functionality is required by the Customer, and IO_Priority shows which IO information how much qualifies Customer's requirements. The Fig. 14 shows information about the graphics component which has been filled by Customer C1.



	Component_id	Funct_id	F_Priority	IO_id	IO_Priority	Cust_id
1	G1	G_1	1	G_IO1	0.9	C1
2	G3	G_1	1	G_IO1	0.9	C1
3	G4	G_1	1	G_IO20	0.3	C1
4	G2	G_1	1	G_IO13	0.9	C1
5	G1	G_10	0.8	G_IO10	0.6	C1
6	G2	G_10	0.8	G_IO16	1	C1
7	G3	G_10	0.8	G_IO10	0.6	C1
8	G3	G_11	0.8	G_IO11	0.8	C1
9	G1	G_11	0.8	G_IO11	0.8	C1
10	G2	G_11	0.8	G_IO11	0.8	C1
11	G1	G_12	0.6	G_IO12	0.9	C1
12	G2	G_13	1	G_IO17	1	C1
13	G2	G_14	1	G_IO18	0.8	C1
14	G4	G_15	0.3	G_IO26	0.9	C1
15	G4	G_16	0.4	G_IO21	0.8	C1
16	G4	G_17	0	G_IO22	0.7	C1
17	G4	G_18	0.6	G_IO23	0.4	C1
18	G4	G_19	0.5	G_IO25	0.8	C1
19	G2	G_2	1	G_IO2	1	C1
20	G4	G_2	1	G_IO2	1	C1
21	G1	G_2	1	G_IO2	1	C1
22	G3	G_2	1	G_IO2	1	C1
23	G4	G_20	0	G_IO24	0.7	C1
24	G3	G_21	0.4	G_IO19	0.2	C1
25	G3	G_3	1	G_IO3	0.6	C1
26	G1	G_3	1	G_IO3	0.6	C1
27	G2	G_3	1	G_IO14	1	C1

Fig. 14 Component_Funct_IO_Priority Table

After changing the values of the priorities and submitting it the component's qualification is determined by applying equation 1based on Customer's needs. The equation will retrieve qualification values for every component as shown in Table 1 at last page of the article.

G2 Qualifies Customer's Requirements up to

75.53846154 %

G3 Qualifies Customer's Requirements up to

59.92307692 %

G1 Qualifies Customer's Requirements up to

59.53846154 %

G4 Qualifies Customer's Requirements up to

19.46153846 %

The bold highlighted **75.53846154** % shows that G2 is the best qualifying component for Customer 'C1'.

6. CONCLUSION

SRSCS approach provides a single repository where components from all other reuse libraries are *Extracted*, *Transformed* and *Loaded*. Components in the SRSCS library are from different *Owner Organizations* and every Owner Organization has their own way of describing the reusable components. If *Customers* want to select their best qualifying component based on functional requirements then a components' comparison is required in the SRSCS library, but the problem is that components are described in different formats and it is not possible to compare them easily. To solve this problem SRSCS provides the Transformation technique that transforms the functional requirements in order to remove redundant information and to classify the components based on component's type. SRSCS not only efficient in term of time by providing all the components in a single repository, but also fully involve the Customer during the Components selection and provides accurate result based on Customer's choice.

7. ACKNOWLEDGEMENT

We are here thankful to International Islamic University which has provided us all the facilities in the completion of this research. We confer special thanks to Assistant Professor Mr. Shahbaz Ahmed Khan who has helped us at each and every point when we faced difficulty.

REFRENCES:

[1] A. Mili , R. Mili and R. Mittermeir , "Storing and Retrieving Software Components: A Refinement Based System", IEEE Trans. Software Eng., vol. 23, no. 7, pp. 445–460, July 1997.

[2] Bernd Fischer, "Specification Based Browsing Of Software Component Libraries", Proceedings of Technical University of Braunschweig, Germany, 1999

[3] Chapter 1. Introduction to IEEE Std. 1517—Software Reuse Processes

[4] Edward A. Boyno, 2003, "Extraction, Transformation, and Loading in a Data Warehouse Course" Proceeding ISECON 2003

[5] Gerald C. Gannod_, Yonghao Chen, and Betty H. C. Cheng , "An Automated Approach for Supporting Software Reuse via Reverse Engineering", 13th IEEE International Conference on Automated Software Engineering (ASE'98), 1998

[6] Haining Yao , Letha Etzkorn, "Towards A Semantic-based Approach for Software Reusable Component Classification and Retrieval", ACM Southeast Regional Conference, 2004

[7] Hyon J. Lee, "Software Reusability", 1998

[8] Jeffrey S. Poulin, Keith .J. Werkman, "Melding Structured Abstracts and the World Wide Web: for Retrieval of Reusable Components", Symposium on Software Reusability,1995

[9] Jo Woodison , Mandarin Consulting, "Managing Software Reuse with Perforce", 2003

[10] Maxym Sjachyn, Ljerka Beus-Dukic, "Semantic Component Selection – SemaCS", 2006

[11] Roger S. Pressman , "Software Engineering", Fifth Edition

[12] Stephen White, Michael Edwards, "DomainEngineering: The challenge, Status, and Trends", 1996

[13] Thomas Jell,"Component Based Software Engineering", CUC 96

[14] Vijayan Sugumaran, Mohan Tanniru and Veda C.Storey, "Identifying software components from process requirements using domain model and object libraries", 1999

[15] Vijayan Sugumaran, Veda C. Storey, "A Semantic-Based Approach to Component Retrieval", 2003

[16] Yonghao Chen, Betty H. C. Cheng, "Formalizing and Automating Component Reuse", 1997

[17] http://www.wright.edu/~luo.3/ncrpdf/ MiniHowTo MultiLoad.pdf

[18] http://www.teradataforum.com/teradata_pdf/ b035-2410- 062a.pdf

[19] http://www.sas.com/offices/europe/czech/ technologies/enterprise_intelligence_platform/ Metagroup_ETL_market.pdf

[20] http://www.1keydata.com/datawarehousing/tooletl.html

[21]http://www.neodynamic.com/Products/
BCWinC/BarcodeWinControl.aspx?tabid=23&
prodid=3&gclid=CMjU_qGUj5QCFSEbagods
VIEfw

[22]http://www.barcodelib.com/java_barcode/
main.html?gclid=CMKNoYeUj5QCFR4vagod
GhsMeQ

[23] http://www.devdirect.com/all/aspose
barcodeforjava_PROD_00017222.aspx

Table 1: Component's Qualification information

Component_id	Funct_id	F_Priority	IO_id	IO_Priority	Customer_id	Qualification of Each Functionality FPi * IOi	$\Sigma_{i=1…N}$ (FPi * IOi)	$Q_c = \Sigma_{i=1…N}$ (FPi * IOi)/ T * 100
G1	G_1	1	G_IO1	0.9	C1	0.9		
G1	G_10	0.8	G_IO10	0.6	C1	0.48		
G1	G_11	0.8	G_IO11	0.8	C1	0.64		
G1	G_12	0.6	G_IO12	0.9	C1	0.54		N=12 T=13
G1	G_2	1	G_IO2	1	C1	1		
G1	G_3	1	G_IO3	0.6	C1	0.6		
G1	G_4	0.8	G_IO4	0.7	C1	0.56		
G1	G_5	0.6	G_IO5	0.9	C1	0.54		
G1	G_6	1	G_IO6	0.8	C1	0.8		
G1	G_7	0.6	G_IO7	0.9	C1	0.54		
G1	G_8	0.9	G_IO8	0.8	C1	0.72		
G1	G_9	0.7	G_IO9	0.3	C1	0.21	**7.74**	**59.53846154**
G2	G_1	1	G_IO13	0.9	C1	0.9		
G2	G_10	0.8	G_IO16	1	C1	0.8		
G2	G_11	0.8	G_IO11	0.8	C1	0.64		N=13 T=13
G2	G_13	1	G_IO17	1	C1	1		
G2	G_14	1	G_IO18	0.8	C1	0.8		
G2	G_2	1	G_IO2	1	C1	1		
G2	G_3	1	G_IO14	1	C1	1		
G2	G_4	0.8	G_IO4	0.7	C1	0.56		
G2	G_5	0.6	G_IO5	0.9	C1	0.54		
G2	G_6	1	G_IO15	0.9	C1	0.9		
G2	G_7	0.6	G_IO7	0.9	C1	0.54		
G2	G_8	0.9	G_IO18	0.8	C1	0.72		
G2	G_9	0.7	G_IO9	0.3	C1	0.21	9.82	**75.53846154**
G3	G_1	1	G_IO1	0.9	C1	0.9		
G3	G_10	0.8	G_IO10	0.6	C1	0.48		
G3	G_11	0.8	G_IO11	0.8	C1	0.64		N=12 T=13
G3	G_2	1	G_IO2	1	C1	1		
G3	G_21	0.4	G_IO19	0.2	C1	0.08		
G3	G_3	1	G_IO3	0.6	C1	0.6		
G3	G_4	0.8	G_IO4	0.7	C1	0.56		
G3	G_5	0.6	G_IO5	0.9	C1	0.54		
G3	G_6	1	G_IO6	0.8	C1	0.8		
G3	G_7	0.6	G_IO7	0.9	C1	0.54		
G3	G_8	0.9	G_IO8	0.8	C1	0.72	7.79	**59.92307692**
G3	G_9	0.7	G_IO9	0.3	C1	0.21		
G4	G_1	1	G_IO20	0.3	C1	0.3		
G4	G_15	0.3	G_IO26	0.9	C1	0.27		
G4	G_16	0.4	G_IO21	0.8	C1	0.32		N=8 T=13
G4	G_17	0	G_IO22	0.7	C1	0		
G4	G_18	0.6	G_IO23	0.4	C1	0.24		
G4	G_19	0.5	G_IO25	0.8	C1	0.4		
G4	G_2	1	G_IO2	1	C1	1		
G4	G_20	0	G_IO24	0.7	C1	0	2.53	**19.46153846**