|  |
| --- |
| **Software Design Specification** |

Study Design Web Service

Create Date: 11/2/2011

Created By: Uttara Sakhadeo

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Uttara Sakhadeo | 11/16/11 | Updates for database schema, web service definition | 1.1.0 |
| Sarah Kreidler | 2/20/2012 | Migrated domain object descriptions to separate document “SDS Domain Layer” | 1.2.0 |
| Uttara Sakhadeo | 6/20/2012 | Changes in External API section, table design. | 1.3.0 |

Table of Contents

[**Software Design Specification** 1](#_Toc327965822)

[1 Introduction 4](#_Toc327965823)

[1.1 Purpose of this document 4](#_Toc327965824)

[1.2 Scope of the development project 4](#_Toc327965825)

[1.3 Definitions, Acronyms, and Abbreviations 4](#_Toc327965826)

[1.4 References 4](#_Toc327965827)

[1.5 Overview of the document 4](#_Toc327965828)

[2 System Architecture Description 6](#_Toc327965829)

[2.1 Overview of modules / components 6](#_Toc327965830)

[2.2 Current Callback Structure 6](#_Toc327965831)

[3 Module and Component Descriptions 7](#_Toc327965832)

[3.1 Component overview 7](#_Toc327965833)

[3.1.1 Study Design Object 7](#_Toc327965834)

[3.1.2 Matrices 7](#_Toc327965835)

[3.1.3 Lists 8](#_Toc327965836)

[3.1.4 Class objects 10](#_Toc327965837)

[3.2 Database 11](#_Toc327965838)

[3.2.1 Table Design 11](#_Toc327965839)

[3.2.2 Scripts 15](#_Toc327965840)

[3.3 Testing 15](#_Toc327965841)

[3.3.1 Unit Testing 15](#_Toc327965842)

[3.3.2 Integration Testing 15](#_Toc327965843)

[3.4 Study Identifiers 15](#_Toc327965844)

[3.5 Objects in the Study Design Service 16](#_Toc327965845)

[3.6 External API 16](#_Toc327965846)

[3.6.1 Objects in the Study Design Service 16](#_Toc327965847)

[3.6.2 The Study Design Resources 16](#_Toc327965848)

[3.6.3 The Beta Scale List Resources 18](#_Toc327965849)

[3.6.4 The Alpha List Resources 19](#_Toc327965850)

[3.6.5 The Relative Group Size List Resources 20](#_Toc327965851)

[3.6.6 The Sample Size List Resources 21](#_Toc327965852)

[3.6.7 The Sigma Scale List Resources 22](#_Toc327965853)

[3.6.8 The Statistical Test List Resources 22](#_Toc327965854)

[3.6.9 The Power Method List Resources 23](#_Toc327965855)

[3.6.10 The Quantile List Resources 24](#_Toc327965856)

[3.6.11 The Response List Resources 25](#_Toc327965857)

[3.6.12 The Nominal Power List Resources 26](#_Toc327965858)

[3.6.13 The Confidence Interval Description Resources 27](#_Toc327965859)

[3.6.14 The Power Curve Description Resources 28](#_Toc327965860)

[3.6.15 The Clustering Resources 28](#_Toc327965861)

[3.6.16 The Between Participant Factor Resources 29](#_Toc327965862)

[3.6.17 The Repeated Measures Resources 30](#_Toc327965863)

[3.6.18 The Matrix Resources 31](#_Toc327965864)

[3.6.19 The Covariance Resources 33](#_Toc327965865)

[3.6.20 The Hypothesis Resources 34](#_Toc327965866)

[3.7 Automated Database Cleanup 36](#_Toc327965867)

[4 Reuse and Relationships to Other Products 37](#_Toc327965868)

[4.1 Reuse of existing code 37](#_Toc327965869)

[4.2 Third Party Dependencies 37](#_Toc327965870)

[5 Design Decisions and Tradeoffs 38](#_Toc327965871)

[6 Appendices 39](#_Toc327965872)

[6.1 Diagrams 39](#_Toc327965873)

# Introduction

## Purpose of this document

This document describes an architectural change in the GLIMMPSE system to improve stability. We introduce a new web service called the Study Design Web Service. The service will store study design definitions for users of the GLIMMSE interface, and will contain all information related to a power or sample size calculation. The Study Design Service will simplify communication between different screens in the user interface. This object will replace the existing callback structure in the GLIMMPSE web-based user interface.

The intended audience for this document is the software engineering team and the grant primary investigators.

## Scope of the development project

The project is an internal architectural change for the user interface. No external functionality will be affected.

## Definitions, Acronyms, and Abbreviations

**CRUD** – An acronym for “create, read, update, delete”. These are the main “verbs” for creating and modifying study designs in the Study Design Service.

**Restlet** – a third party framework for implementing Restful web services.

**Hibernate** – a third party framework for mapping Java objects into relational database tables

**EER diagram –** Entity Relationship model is used to produce a type of conceptual schema of a system.

**JSON –** Java Script Object Notation. A Java/C/C++/and many other language objects can be displayed as text format usin JSON.

## References

1. Anon. Restlet 2.0 - Documentation. Available at: http://www.restlet.org/documentation/2.0/. Accessed March 9, 2012.

2. Anon. Extensible Markup Language (XML) 1.0 (Fifth Edition). Available at: http://www.w3.org/TR/REC-xml/. Accessed February 10, 2012.

3. Noelios Technologies. Restlet. Version 1.1.6. 2009. Available at: http://www.restlet.org/. Accessed May 17, 2011.

4. Apache Software Foundation. Apache Tomcat. Version 6.0. Available at: http://tomcat.apache.org/. Accessed May 17, 2011.

5. Sakhadeo U, Kreidler S. Software Design Specification: GLIMMPSE Domain Objects. 2011.

6. Muller KE, Stewart PW. *Linear model theory: univariate, multivariate, and mixed models*. Hoboken, New Jersey: John Wiley and Sons; 2006.

7. Anon. First application. Available at: http://wiki.restlet.org/docs\_2.0/13-restlet/21-restlet/318-restlet/303-restlet.html. Accessed March 9, 2012.

## Overview of the document

The document provides a brief review of the existing callback structure in the GLIMMPSE interface. We define the Study Design Service and describe how it will replace the existing callback structure. We sketch the class structure and database schema for the Study Design Service and changes required for existing user interface.

# System Architecture Description

## Overview of modules / components

The GLIMMPSE web user interface (web-GUI) is the browser-based front end of the GLIMMPSE software system. The web-GUI elicits information from the user about the study design and hypothesis of interest. The study design information is converted into a series of matrices used to calculate power or sample size for the general linear multivariate model. The user interface is responsible for:

1. Collecting information from the user
2. Validating the information entered by the user
3. Converting the inputs into matrices required for a power or sample size calculation
4. Submitting the matrices to the Power Service to perform the calculation
5. Receiving and displaying the results to the user
6. Optionally, the user interface may submit the information to the Chart Service and display the resulting power curve.

## Current Callback Structure

Communication between screens in the GLIMMPSE user interface occurs via several listener classes. Screens send callbacks to one another to synchronize the user experience, but there is no central object containing all information about the overall study design. This design underlies several existing bugs in the user interface. In addition, although users can download and upload study designs from GLIMMPSE website, there isn’t any facility for global storage of study designs.

To solve these problems, GLIMMPSE version 2 will add a web service called the Study Design Service. The Study Design Service will provide a central repository of study design information. User interface screens will call into the Study Design Service to access required information about the current study design. This will eliminate the current callback structure and reduce the potential for screen synchronization bugs.

The service will be implemented as a Restlet, running in a Tomcat server. The service will provide a CRUD API for creating and modifying study designs, which will be stored in a MySQL database. Object relational mapping will be implemented with Hibernate.

# Module and Component Descriptions

## Component overview

Study Design service will provide a container, the Study Design, for holding information in the form of lists, matrices, objects and primitive data types. The Study Design contains the following information:

### Study Design Object

The study design object includes some basic information about the type of calculation, and maps to a group of matrices, lists, and other meta information related to the study design. The subcomponents of the study design object are described in each section below. The simple fields for the study design object are:

#### Solving For type :

The solving for flag indicates whether the user is solving for power, sample size, or detectable difference. The value will be an ENUM field, coded as follows.

Selection Value

Power 1

Total Sample Size 2

Detectable Difference 3

#### Gaussian Covariate Flag :

This flag indicates that the user wishes to control for a Gaussian covariate in the power calculation. This will be a Boolean field

### Matrices

Several matrices are associated with the Study Design object. Muller and Stewart provides details on matrix definitions for the univariate and multivariate linear model. The matrices which can be stored with a study design object are listed in the table below

|  |  |  |  |
| --- | --- | --- | --- |
| **Matrix** | **Rows** | **Columns** | **Conformance Details** |
| X: Design essence matrix | N | Q | Columns equal number of rows in β |
| C: Between subject contrast | A | Q | Columns equal number of rows in β  Rows must be one less than number of rows in X |
| U: Within subject contrast | P | B | Rows equals the number of columns in β |
| Θ0: Null hypothesis values | A | B | Rows equals the number of rows in C Columns equals the number of columns in U |
| β: Regression coefficient estimates | Q | P | Rows equals the number of columns in X (number of predictors) Columns equals the number of columns in U (number of outcomes) |
| **GLMM(F) Designs** | | | |
| Σerror: covariance matrix of residuals | P | P | Matrix is square and symmetric. Rows/columns equal the number of columns in β |
| **GLMM(F,g) Designs** | | | |
| ΣG: Covariance of Gaussian predictor | 1 | 1 | Since only a single baseline covariate is allowed, this matrix is 1x1. |
| ΣYG: Covariance of Gaussian predictor with outcomes | P | 1 | Rows equals the number of columns in β |
| ΣY: Covariance of outcomes | P | P | Rows/columns equal the number of columns in β (i.e. number of outcomes) |

### Lists

Using these lists users can specify several statistical tests, power methods (for designs with a baseline covariate), power values, alpha levels, and per group sample size values etc.

Users can scale the regression coefficients in the beta matrix and the variability specified in the sigma matrix. The following lists are stored with the Study Design;

#### Alpha List :

The alpha list specifies various type I error rates.

#### Relative Group Size List :

This list tells the application-middle layer whether the groups are of equal size or of unequal sizes by letting users specify ratio between the entities.

Let us consider an example; suppose there are two groups A and B

Case 1: GLIMMPSE allows users to specify the equality by putting value as 1 in each of the text boxes.

Case 2: Inequality is given by ratios for example 4:1 or 2:1 or so on

#### Per Group Sample Size List :

The per group sample size list includes all desired group sample sizes (reference REPN in powerlib). It is valid for power or detectable difference requests (it is ignored for sample size requests). The total sample size for the power calculation is determined by multiplying the per group size by the number of rows in the design matrix.

For example, consider the following design matrix for a one-way ANOVA with 3 groups:

For a per group sample size of 25, the total sample size would be 75.

#### List of Responses :

This list includes different responses taken at the time of each measurement (e.g. Heart Rate, Blood Pressure). The input string should contain letters, numbers, dashes, periods or underscores

Validation rules :

1. The responses string or category string cannot contain following symbols <, >
2. No spaces are allowed in between the words.

#### Beta Scale List :

The beta scale list includes all scale factors for the regression coefficient, or beta, matrix. The beta scale list is valid for power and sample size requests (it is ignored for detectable difference requests).

#### Sigma Scale List :

The sigma scale list includes all scale factors for the error covariance matrix. The sigma scale list is required for all requests.

#### Test List :

The test list includes all statistical tests to be run. It is required for all requests. The test list is specified as:

Power values should be equivalent for univariate designs regardless of the test selected. However, no uniformly most powerful test exists for the multivariate linear model. The following tests are supported:

unirep - Univariate approach to repeated measures (uncorrected)

unirepBox - Univariate approach to repeated measures with Box correction

unirepGG - Univariate approach to repeated measures with Geisser-Greenhouse correction

unirepHF - Univariate approach to repeated measures with Huynh-Feldt correction

wl - Wilks' Lambda

hlt - Hotelling-Lawley trace

pbt - Pillai-Bartlett trace

#### Power Method List:

The power method lists includes any combination of the conditional, unconditional, or quantile power methods. It is valid only for study designs involving a baseline covariate (ignored otherwise). Note that only the Hotelling-Lawley and univariate approach to repeated measures are supported for designs with a baseline covariate.

The following power methods are supported:

conditional - Power values conditional on the specific realization of the baseline covariate values

unconditional - Power based on numerical integration over possible values of the non-centrality parameter

quantile - Power based on specific quantiles of the non-centrality parameter's distribution

#### Quantile List :

The quantile list specifies all quantiles for use with the "quantile" power method. This list is valid only for study designs with a baseline covariate, which specify quantile power in the power method list.

### Class objects

For Matrix Mode, only the matrices need to be stored with the study design. For Guided Mode, uses specify additional information about predictor and response names, hypotheses, etc. For study designs produced by Guided Mode, the following objects are stored with the Study Design;

#### Object describing Between Subject Effect:

Between subject effects describe fixed predictors and the associated list of possible values. This object contains following fields.

|  |  |  |
| --- | --- | --- |
| **Variable** | **Type** | **Description** |
| Predictor name | String | Name of the between participant factor |
| Category list | List | Names of all valid categories for this factor |

This object is required for producing the X matrix. For Guided Mode, both the between subject effect information and associated X matrix will be stored in the database.

#### Object describing Within Subject Effect:

This object contains following details.

1. Information about repeated measures.
2. Information regarding clustering.

See the “Software Design Specification: Multilevel and Longitudinal Study Designs” for full details on within subject effect information

#### Hypothesis Object:

This object contains information regarding the primary hypothesis of interest. Hypothesis objects describe the type of hypothesis, and all variables associated with the hypothesis.

See the “Software Design Specification: Full Factor Hypothesis Selection” for full details on hypothesis specification.

#### Covariance Information

The Study Design object will store information regarding the covariance for multilevel and longitudinal designs.

Covariance information can be specified either as structured or unstructured, and parameterized by covariance or correlation plus standard deviation. For matrix mode, only the sigma matrices will be stored. For guided mode, Lear model information for structured matrices will be stored.

See the “Software Design Specification: Complex Covariance Specification” for full details on hypothesis specification

#### Power Curve description Object:

This object holds information required for drawing power curve. This includes the following information:

|  |  |  |
| --- | --- | --- |
| **Variable** | **Type** | **Description** |
| XAxis Type | String | Variable to display on the X-axis |
| Stratification Variable | String | Displays multiple curves based on the stratification variable |
| Remaining Variable Values | List | List of values which are fixed for the remaining variables |

#### Confidence Interval Object:

This object describes the type of confidence interval to produce around each power value. It contains the following variable instances;

|  |  |  |
| --- | --- | --- |
| **Variable** | **Type** | **Description** |
| BetaFixed | Boolean | Indicates if the beta matrix is assumed fixed |
| SigmaFixed | Boolean | Indicates weather Sigma matrix is fixed |
| LowerTailProbability | Double | Lower tail probability for the confidence interval |
| UpperTailProbability | Double | Upper tail probability for the confidence interval |
| Sample Size | Int | Sample size of the data from which the beta and sigma values were obtained |
| Design matrix rank | Int | Rank of the design matrix for the model from which the beta and sigma values were obtained |

## Database

### Table Design

All the matrices, lists and objects will be stored in proper data base tables. The Database Table design is available in the **StudyDesignService\_version\_1\_0\_0\_databaseSchema.mwb** file. See the attached PDF document for a global view of the database schema.

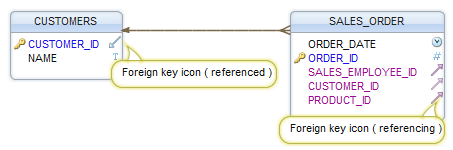
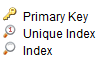
**ERR diagram description:**



Two related entities shown using Crow's Foot notation. In this example, an optional relationship is shown between Artist and Song; the symbols closest to the song entity represents "zero, one, or many", whereas a song has "one and only one" Artist. The former is therefore read as, an Artist (can) perform(s) "zero, one, or many" song(s).

**Symbols Used :**

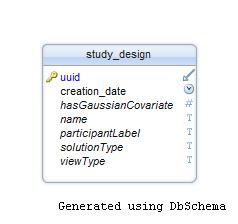
1. Keys

1. Relationships

 One to one Relationship  MO (Many to One) Relationship

#### Study Design



**Figure 1 : ‘Study\_Design’ table.**

Study design table uses UUID, (refer section 3.4) as primary identifier. This UUID for a study design is referred from other children tables as foreign key reference.

#### Alpha List

There is MO-AS relationship between Alpha values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from alpha\_list table are related to one Study\_design record.
* AS (All to Some): Each alpha list table record is connected to a study design table record. But a study design table record can exist with one/many or without any alpha list record/s connected to it.

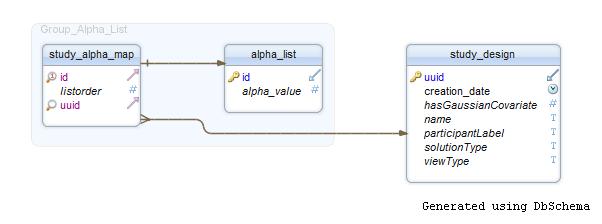


Figure 2 : MOAS Relationship between ‘Alpha\_List’ and ‘Study\_Design’ tables.

#### Beta Scale List

There is MO-AS relationship between Beta Scale values and Study Design.

MO-AS relationship:

* MO (Many to One) : Many values from beta\_scale\_list table are related to one Study\_design
* AS (All to Some): Each beta scale list table record is connected to a study design table record. But a study design table record can exist with one/many or without any beta scale list record/s connected to it.

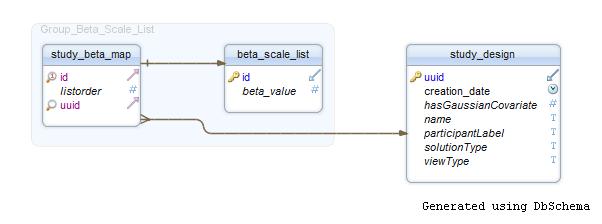


Figure 3 : MOAS Relationship between ‘Beta\_Scale\_List’ and ‘Study\_Design’ tables.

#### Sigma Scale List

There is MO-AS relationship between Sigma Scale and Study Design values.

MO-AS relationship:

* MO (Many to One) : Many values from sigma\_scale\_list table are related to one Study\_design
* AS (All to Some): Each sigma scale list table record is connected to a study design table record. But a study design table record can exist with one/many or without any sigma scale list record/s connected to it.

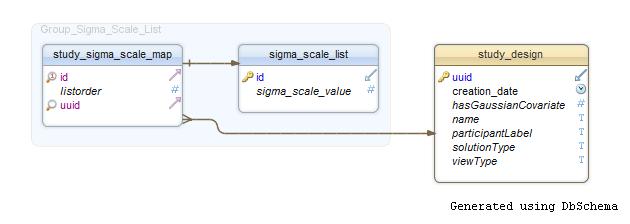


Figure 4 : MOAS Relationship between ‘Sigma\_Scale\_List’ and ‘Study\_Design’ tables.

#### Nominal Power List

There is MO-AS relationship between Nominal Power values and Study Design.

MO-AS relationship:

* MO (Many to One) : Many values from nominal\_power\_list table are related to one Study\_design
* AS (All to Some): Each nominal power list table record is connected to a study design table record. But a study design table record can exist with one/many or without any nominal power list record/s connected to it.

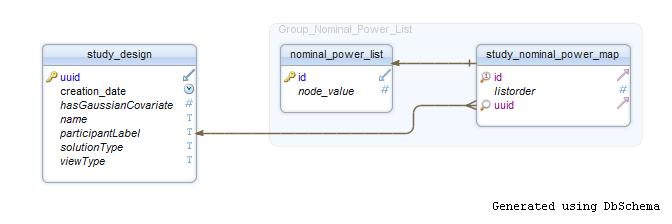


Figure 5 : MOAS Relationship between ‘Nominal\_Power\_List’ and ‘Study\_Design’ tables.

#### Quantile List

There is MO-AS relationship between Quantile values and Study Design.

MO-AS relationship:

* MO (Many to One) : Many values from Quantile\_list table are related to one Study\_design
* AS (All to Some) : Each quantile list table record is connected to a study design table record. But a study design table record can exist with one/many or without any quantile list record/s connected to it.

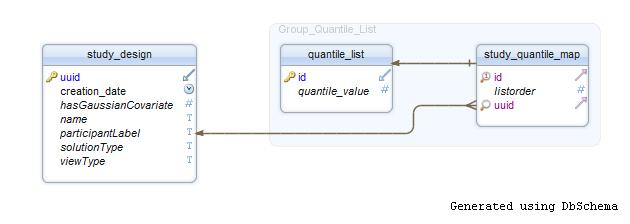


Figure 6 : MOAS Relationship between ‘Quantile\_List’ and ‘Study\_Design’ tables.

#### Relative Group Size List

There is MO-AS relationship between Relative Group Size values and Study Desig.

MO-AS relationship:

* MO (Many to One) : Many values from relative\_group\_size\_list table are related to one Study\_design
* AS (All to Some): Each relative group size list table record is connected to a study design table record. But a study design table record can exist with one/many or without any relative group size list record/s connected to it.

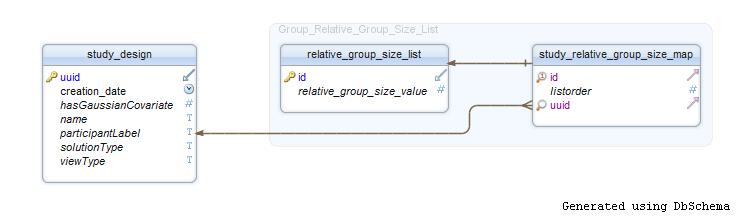


Figure 7 : MOAS Relationship between ‘Relative\_Group\_Size\_List’ and ‘Study\_Design’ tables.

#### Power Method List

There is MO-AS relationship between Power Method values and Study Design.

MO-AS relationship:

* MO (Many to One) : Many values from power\_method\_list table are related to one Study\_design
* AS (All to Some): Each power method list table record is connected to a study design table record. But a study design table record can exist with one/many or without any power method list record/s connected to it.

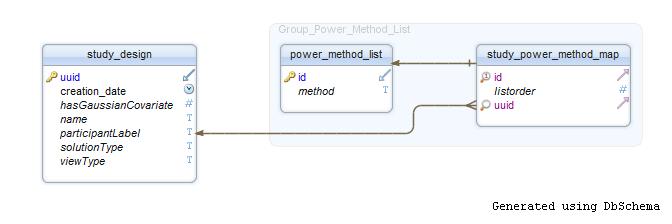


Figure 8 : MOAS Relationship between ‘Power\_Method\_List’ and ‘Study\_Design’ tables.

#### Response List

There is MO-AS relationship between Response values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from response\_list table are related to one Study\_design
* AS (All to Some): Each response list table record is connected to a study design table record. But a study design table record can exist with one/many or without any response list record/s connected to it.

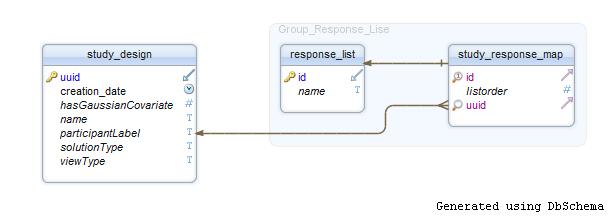


Figure 9 : MOAS Relationship between ‘Response\_List’ and ‘Study\_Design’ tables.

#### Sample Size List

There is MO-AS relationship between Sample Size values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from sample\_size\_list table are related to one Study\_design
* AS (All to Some): Each nominal sample size list table record is connected to a study design table record. But a study design table record can exist with one/many or without any sample size list record/s connected to it.

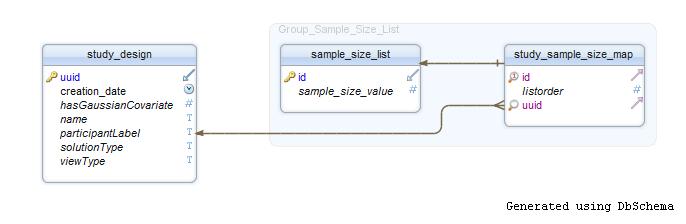


Figure 10 : MOAS Relationship between ‘Sample\_Size\_List’ and ‘Study\_Design’ tables.

#### Statistical Test List

There is MO-AS relationship between Statistical Test values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from statistical\_test\_list table are related to one Study\_design
* AS (All to Some): Each statistical test list table record is connected to a study design table record. But a study design table record can exist with one/many or without any statistical test list record/s connected to it.

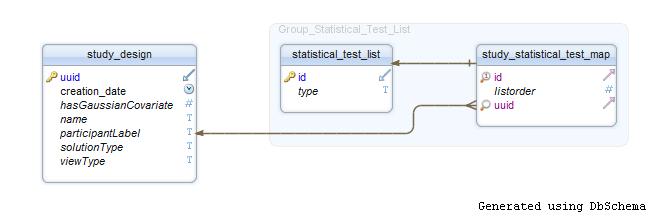


Figure 11 : MOAS Relationship between ‘Statistical\_Test\_List’ and ‘Study\_Design’ tables.

#### Confidence Interval Description

There is OO-AS relationship between Study Design and Confidence Interval Description values.

OO-AS relationship:

* OO (One to One): Only one value from table Confidence Interval Description is related to one Study\_design
* AS (All to Some): Each Confidence Interval Description table record is connected to a study design table record. But a study design record can exist with/without a Confidence Interval Description record connected to it.

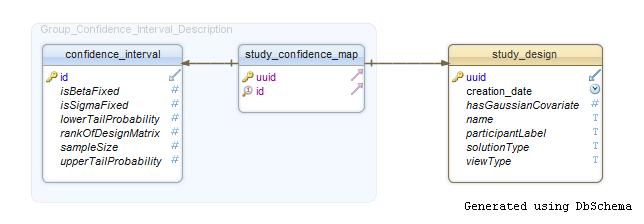


Figure 12 : OOAS Relationship between ‘Confidence Interval Description’ and ‘Study\_Design’ tables.

#### Power Curve Description

There is OO-AS relationship between Study Design and Power Curve Description values.

OO-AS relationship:

* OO (One to One): Only one value from table Power Curve Description is related to one Study\_design
* AS (All to Some: Each Power Curve Description table record is connected to a study design table record. But a study design record can exist with/without a Power Curve Description record connected to it.

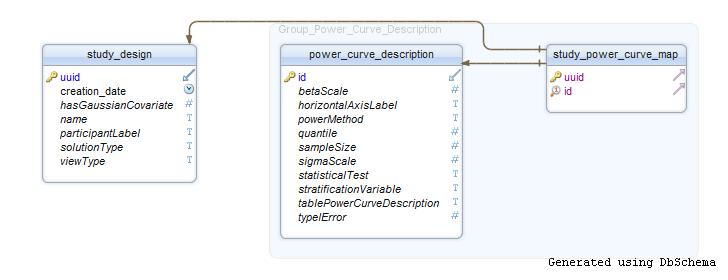


Figure 13 : OOAS Relationship between ‘Power Curve Description’ and ‘Study\_Design’ tables.

#### Matrix

There is MO-AS relationship between Matrix table values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from matrix table are related to one Study\_design record.
* AS (All to Some: Each matrix table record is connected to a study design table record. But a study design table record can exist with one/many or without any matrix record/s connected to it.

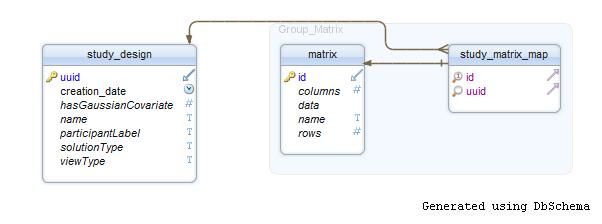


Figure 14 : MOAS Relationship between ‘Matrix’ and ‘Study\_Design’ tables.

#### Covariance

There is MO-AS relationship between Covariance table values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from covariance table are related to one Study\_design record.
* AS (All to Some): Each covariance table record is connected to a study design table record. But a study design table record can exist with one/many or without any covariance record/s connected to it.
* There is MO-AS relationship between Standard Deviation table and Covariance table too.

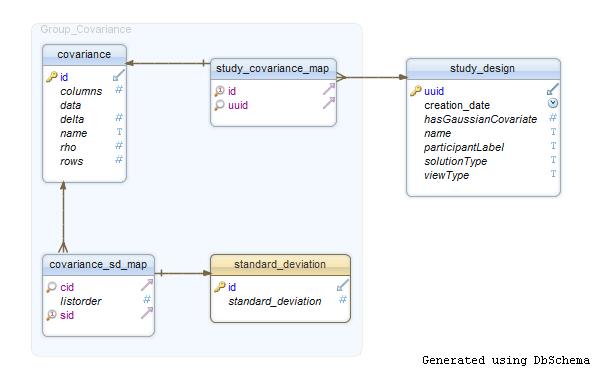


Figure 15 : MOAS Relationship between ‘Covariance’ and ‘Study\_Design’ tables.

#### Clustering

There is MO-AS relationship between Cluster Node values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from Cluster Node table are related to one Study\_design record.
* AS (All to Some): Each Cluster Node table record is connected to a study design table record. But a study design table record can exist with one/many or without any Cluster Node record/s connected to it.

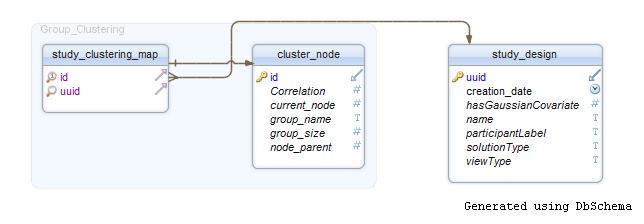


Figure 16 : MOAS Relationship between ‘Cluster\_Node’ and ‘Study\_Design’ tables.

#### Between Participant Factors

There is MO-AS relationship Between Participant effects values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from Between Participant effects table are related to one Study design table record.
* AS (All to Some): Each Between Participant effects table record is connected to a study design table record. But a study design table record can exist with one/many or without any Between Participant effects record/s connected to it.

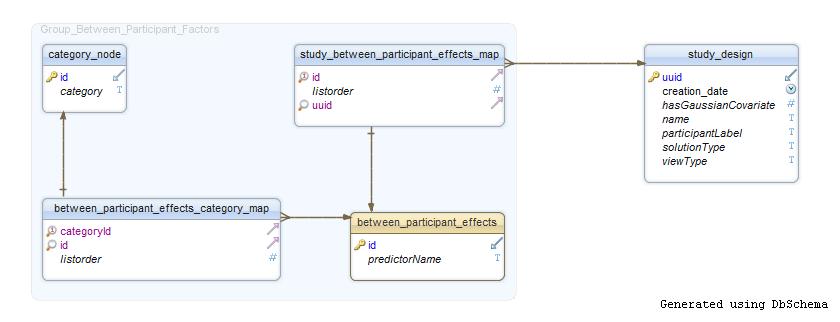


Figure 17 : MOAS Relationship between ‘Between\_Participant\_Effects’ and ‘Study\_Design’ tables.

#### Repeated Measures

There is MO-AS relationship between Repeated Measures values and Study Design.

MO-AS relationship:

* MO (Many to One): Many values from Repeated Measures table are related to one Study design record.
* AS (All to Some): Each Repeated Measures table record is connected to a study design table record. But a study design table record can exist with one/many or without any Repeated Measures record/s connected to it.

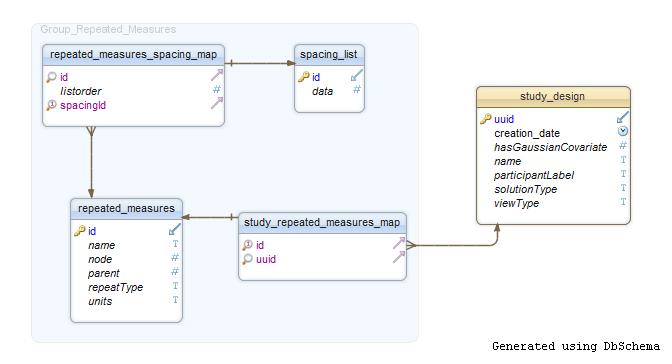


Figure 18 : MOAS Relationship between ‘Repeated\_Measures’ and ‘Study\_Design’ tables.

#### Hypothesis

There is MO-AS relationship between Hypothesis values and Study Design.

MO-AS relationship :

* MO (Many to One): Many values from Hypothesis table are related to one Study\_design record.
* AS (All to Some): Each Hypothesis table record is connected to a study design table record. But a study design table record can exist with one/many or without any Hypothesis record/s connected to it.

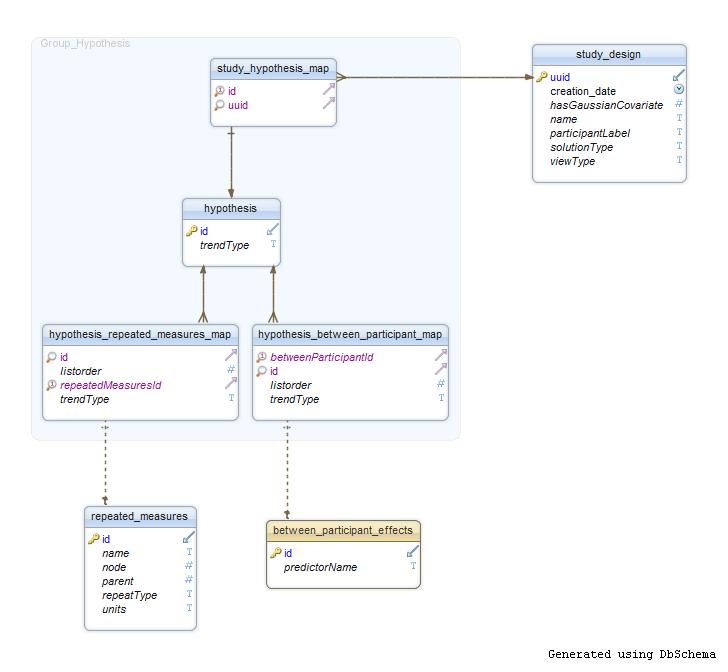


Figure 19 : MOAS Relationship between ‘Hypothesis’ and ‘Study\_Design’ tables. MOSS Relationship between ‘Repeated\_Measures’ and ‘Hypothesis’ tables. OOSS Relationship between ‘Between\_Participant\_Effects’ and ‘Hypothesis’ tables

MO-SS relationship between ‘Repeated\_Measures’ and ‘Hypothesis’ tables:

* MO (Many to One): Many values from Repeated Measures table are related to one Hypothesis record.
* SS (Some to Some): For a study design, Hypothesis record can exist independent of Repeated Measures values. Similarly Repeated Measures values can exist independent of Hypothesis record, for a study design.

MO-SS relationship between ‘Between\_Participant\_Effects’ and ‘Hypothesis’ tables:

* MO (Many to One): Many values from Between Participant Effects table are related to one Hypothesis record.
* SS (Some to Some): For a study design, Hypothesis record can exist independent of Between Participant Effects values. Similarly Between Participant Effects values can exist independent of Hypothesis record, for a study design.

### Scripts

Scripts will be provided to perform the following database activities:

* Creation of a new database and all associated tables
* A script to remove expired study designs from the database
* Creation of erase history function.

## Testing

### Unit Testing

Here newly created modules will be tested for against their functionality. This testing will be done as per predefined test cases (e.g. for functions etc.)

#### Unit tests for Hibernate

With the help of JUnit data access layer can be tested. Hibernate has lots of files, hence unit testing for each of such file is required. Here are some of the file types used in Hibernate

1. Persistent (POJO) classes – Plain Old Java Object classes are used to map to database table.
2. Hibernate Mapping file (hbm.xml) – This file maps POJO class object with corresponding table.
3. Hibernate Configuration File(hibernate.cfg.xml) – This file is used to create connection pool and to setup required environment.
4. Java files for accessing data from tables – Actual queries will be written here.

#### Unit tests for Protocols

Protocols like xml parsing are also needed to be tested. Some existing methodology for such a unit testing will be reused here.

### Integration Testing

All the separate modules after integrating together will be tested to assure that required functionality is provided.

## Study Identifiers

When a user creates a new study design, a UUID will be assigned to it. The study design UUID will be used to synchronize the overall study design across multiple database tables. The GLIMMPSE UI screens will retrieve study design information via the study UUID. Users can retrieve stored study designs by providing this UUID.

Structure of UUID:

In general UUIDs are either 16-byte (128-bit) or 32-byte long.

Here are some examples of UUIDs:

067e6162-3b6f-4ae2-a171-2470b63dff00

54947df8-0e9e-4471-a2f9-9af509fb5889

Expected Problems:

Now there are two important issues with such a UUID based design;

1. User has to remember each of such UUID. This information may be stored as session cookies. In subsequent versions, these may be associated with a user email or login/password combination.
2. Any user can access any study design. This can lead to a problem of security of such study designs.

UUID duplication probability:

These UUIDs are supposed to be ‘practically unique’ rather than ‘guaranteed unique’.

There are 16^32 = 340,282,366,920,938,463,463,374,607,431,768,211,456 possible UUIDs

Thus the probability of creating a few tens of trillions of UUIDs in a year and having one duplicate is 0.00000000006

Note:

Each created study design will be removed from data base after a month with erase all history option.

## Objects in the Study Design Service

## External API

Clients can communicate with the server using Restlet Client API. The service accepts serialized Java Objects as input. The returned HTTP request will use the application type

application/x-java-serialized-object.

The API described below allows users to create, read, update and delete study designs and associated components of study designs. As a general rule, the following HTTP protocols are used for each operation

|  |  |
| --- | --- |
| **HTTP Request Type** | **Operation** |
| GET | Read |
| POST | Create |
| PUT | Create or Update |
| DELETE | Delete |

### Objects in the Study Design Service

All the screens will communicate with Study Design Service through JSON. These screens will populate all the information in the objects and then forward these objects using JSON to Study Design Service. Study Design Service then will take these objects and store them in database using Hibernate. Or otherwise Study Design Service will fetch the objects from data base using Hibernate and send back to screens using JSON.

StudyDesign service takes almost each domain object as an input. These objects are discussed in detail in the **SDS\_GLIMMPSE\_DomainLayer\_03.docx** file in detail.

### The Study Design Resources

Each resource interface has a server resource class which implements this interface.

#### The Study Design Resource

The Study Design Resource is the external interface for creating/updating/deleting a Study Design.

URI to call the Study Design Server Resource class is

http://tomcat-server/study/study

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| StudyDesign | Create()  Creates a new Study Design which has only a new uuid associated with it.  Saves this Study Design into database table.  Returns this newly created and saved Study Design. |
| StudyDesign | update(StudyDesign studyDesign)  Updates specified Study Design into database table.  If uuid already present rewrites the existing contents. Otherwise creates a new record.  Returns this newly updated Study Design. |
| StudyDesign | remove(byte[] uuid)  Deletes Study Design with specified uuid from database table.  Returns deleted Study Design. |

#### The Study Design Retrieve Resource

The Study Design Upload Retrieve Resource is the external interface for retrieving a Study Design with specified uuid.

URI to call the Study Design Retrive Server Resource class is

http://tomcat-server/study/study/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| StudyDesign | retrieve(byte[] uuid)  Retrieves and returns the Study Design with the specified uuid from database. |

#### The Upload Resource

The Upload Resource is the external interface for retrieving a Study Design with specified uuid.

URI to call the Upload Server Resource class is

http://tomcat-server/study/upload

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| Representation | upload(Representation entity)  Uploads and saves/updates provided Study Design file into the database. This saved Study Design is then returned to the user. |

#### The SaveAs Resource

The Study Design Upload Retrieve Resource is the external interface for retrieving a Study Design with specified uuid.

URI to call the SaveAs Server Resource class is

http://tomcat-server/study/saveas

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| Representation | saveAs(Representation entity)  Retrieves the Study Design with specified uuid from database. This Study Design is saved on Disk as a json file. |

#### The Study Design Upload Retrieve Resource

The Study Design Upload Retrieve Resource is the external interface for retrieving a Study Design with specified uuid.

URI to call the Study Design Upload Retrieve Server Resource class is

http://tomcat-server/study/studyUploadRetrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| StudyDesign | upload(StudyDesign studyDesign)  Uploads the Study Design with specified uuid into the database. |
| StudyDesignList | retrieve()  Retrieves list of Study Designs stored into the database. |

### The Beta Scale List Resources

#### The Beta Scale Resource

The Beta Scale Resource is the external interface for creating/updating/deleting list of Beta Scale objects. BetaScaleList object is a domain layer class, for more details please refer section 3.1.30 of Domain Layer Specification Document.

URI to call the Beta Scale Server Resource class is

http://tomcat-server/study/betaScaleList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| BetaScaleList | Create(BetaScaleList list)  Creates, saves and returns a list of beta scale objects associated with specified uuid. |
| BetaScaleList | update(BetaScaleList list)  Updates the existing list of beta scale objects with specified list for a given uuid.  Returns this newly updated List of beta scales. |
| BetaScaleList | remove(byte[] uuid)  Deletes and returns List of beta scale objects associated with specified uuid from database table. |

#### The Beta Scale Retrieve Resource

The Beta Scale Retrieve Resource is the external interface for retrieving a list of beta scale objects associated with specified uuid.

URI to call the Beta Scale Retrieve Server Resource class is

http://tomcat-server/study/betaScaleList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| BetaScaleList | retrieve(byte[] uuid)  Retrieves and returns list of beta scale objects associated with the specified uuid from database. |

### The Alpha List Resources

#### The Type I Error Resource

The Type I Error Resource is the external interface for creating/updating/deleting list of Type I Error objects. TypeIErrorList object is a domain layer class, for more details please refer section 3.1.28 of Domain Layer Specification Document.

URI to call the Type I Error Server Resource class is

http://tomcat-server/study/alphaList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| TypeIErrorList | Create(TypeIErrorList list)  Creates, saves and returns a list of type I error objects associated with specified uuid. |
| TypeIErrorList | update(TypeIErrorList list)  Updates the existing list of type I Error objects with specified list for a given uuid.  Returns this newly updated List of type I error objects. |
| TypeIErrorList | remove(byte[] uuid)  Deletes and returns a completeList of type I error objects associated with specified uuid from database table. |

#### The Type I Error Retrieve Resource

The Type I Error Retrieve Resource is the external interface for retrieving a list of type I error objects associated with specified uuid.

URI to call the Type I Error Retrieve Server Resource class is

http://tomcat-server/study/alphaList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| TypeIErrorList | retrieve(byte[] uuid)  Retrieves and returns a complete list of type I error objects associated with the specified uuid from database. |

### The Relative Group Size List Resources

#### The Relative Group Size Resource

The Relative Group Size Resource is the external interface for creating/updating/deleting list of Relative Group Size objects. RelativeGruopSizeList object is a domain layer class, for more details please refer section 3.1.34 of Domain Layer Specification Document.

URI to call the Relative Group Size Server Resource class is

http://tomcat-server/study/relativeGroupSizeList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| RelativeGroupSizeList | Create(RelativeGroupSizeList list)  Creates, saves and returns a list of relative group size objects associated with specified uuid. |
| RelativeGroupSizeList | update(RelativeGroupSizeList list)  Updates the existing list of relative group size objects with specified list for a given uuid.  Returns this newly updated List of relative group size objects. |
| RelativeGroupSizeList | remove(byte[] uuid)  Deletes and returns a complete List of relative group size objects associated with specified uuid from database table. |

#### The Relative Group Size Retrieve Resource

The Relative Group Size Retrieve Resource is the external interface for retrieving a list of relative group size objects associated with specified uuid.

URI to call the Relative Group Size Retrieve Server Resource class is

http://tomcat-server/study/relativeGroupSizeList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| RelativeGroupSizeList | retrieve(byte[] uuid)  Retrieves and returns a complete list of relative group size objects associated with the specified uuid from database. |

### The Sample Size List Resources

#### The Sample Size Resource

The Sample Size Resource is the external interface for creating/updating/deleting list of Sample Size objects. SampleSizeList object is a domain layer class, for more details please refer section 3.1.46 of Domain Layer Specification Document.

URI to call the Sample Size Server Resource class is

http://tomcat-server/study/sampleSizeList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| SampleSizeList | Create(RelativeGroupSizeList list)  Creates, saves and returns a list of sample size objects associated with specified uuid. |
| SampleSizeList | update(RelativeGroupSizeList list)  Updates the existing list of sample size objects with specified list for a given uuid.  Returns this newly updated List of sample size objects. |
| SampleSizeList | remove(byte[] uuid)  Deletes and returns a complete List of sample size objects associated with specified uuid from database table. |

#### The Relative Group Size Retrieve Resource

The Sample Size Retrieve Resource is the external interface for retrieving a list of sample size objects associated with specified uuid.

URI to call the Sample Size Retrieve Server Resource class is

http://tomcat-server/study/sampleSizeList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| SampleSizeList | retrieve(byte[] uuid)  Retrieves and returns a complete list of sample size objects associated with the specified uuid from database. |

### The Sigma Scale List Resources

#### The Sigma Scale Resource

The Sigma Scale Resource is the external interface for creating/updating/deleting list of Sigma Scale objects. SigmaScaleList object is a domain layer class, for more details please refer section 3.1.32 of Domain Layer Specification Document.

URI to call the Sigma Scale Server Resource class is

http://tomcat-server/study/sigmaScaleList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| SigmaScaleList | Create(SigmaScaleList list)  Creates, saves and returns a list of sigma scale objects associated with specified uuid. |
| SigmaScaleList | update(SigmaScaleList list)  Updates the existing list of sigma scale objects with specified list for a given uuid.  Returns this newly updated List of sigma scale objects. |
| SigmaScaleList | remove(byte[] uuid)  Deletes and returns a complete List of sigma scale objects associated with specified uuid from database table. |

#### The Sigma Scale Retrieve Resource

The Sigma Scale Retrieve Resource is the external interface for retrieving a list of Sigma Scale objects associated with specified uuid.

URI to call the Sigma Scale Retrieve Server Resource class is

http://tomcat-server/study/sigmaScaleList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| SigmaScaleList | retrieve(byte[] uuid)  Retrieves and returns a complete list of sigma scale objects associated with the specified uuid from database. |

### The Statistical Test List Resources

#### The Statistical Test Resource

The Statistical Test Resource is the external interface for creating/updating/deleting list of Statistical Test objects. StatisticalTestList object is a domain layer class, for more details please refer section 3.1.36 of Domain Layer Specification Document.

URI to call the Statistical Test Server Resource class is

http://tomcat-server/study/testList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| StatisticalTestList | Create(StatisticalTestList list)  Creates, saves and returns a list of Statistical Test objects associated with specified uuid. |
| StatisticalTestList | update(StatisticalTestList list)  Updates the existing list of Statistical Test objects with specified list for a given uuid.  Returns this newly updated the complete List of Statistical Test objects. |
| StatisticalTestList | remove(byte[] uuid)  Deletes and returns a complete List of Statistical Test objects associated with specified uuid from database table. |

#### The Statistical Test Retrieve Resource

The Statistical Test Retrieve Resource is the external interface for retrieving a list of Statistical Test objects associated with specified uuid.

URI to call the Statistical Test Retrieve Server Resource class is

http://tomcat-server/study/testList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| StatisticalTestList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Statistical Test objects associated with the specified uuid from database. |

### The Power Method List Resources

#### The Power Method Resource

The Power Method Resource is the external interface for creating/updating/deleting list of Power Method objects. PowerMethodList object is a domain layer class, for more details please refer section 3.1.38 of Domain Layer Specification Document.

URI to call the Power Method Server Resource class is

http://tomcat-server/study/powerMethodList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| PowerMethodList | Create(PowerMethodList list)  Creates, saves and returns a list of Power Method objects associated with specified uuid. |
| PowerMethodList | update(PowerMethodList list)  Updates the existing list of Power Mtehod objects with specified list for a given uuid.  Returns this newly updated List of Power Method objects. |
| PowerMethodList | remove(byte[] uuid)  Deletes and returns a complete List of Power Method objects associated with specified uuid from database table. |

#### The Power Method Retrieve Resource

The Power Method Retrieve Resource is the external interface for retrieving a list of Power Method objects associated with specified uuid.

URI to call the Power Method Retrieve Server Resource class is

http://tomcat-server/study/powerMethodList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| PowerMethodList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Power Method objects associated with the specified uuid from database. |

### The Quantile List Resources

#### The Quantil Resource

The Quantile Resource is the external interface for creating/updating/deleting list of Quantile objects. QuantileList object is a domain layer class, for more details please refer section 3.1.40 of Domain Layer Specification Document.

URI to call the Quantile Server Resource class is

http://tomcat-server/study/quantileList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| QuantileList | Create(QuantileList list)  Creates, saves and returns a list of Quantile objects associated with specified uuid. |
| QuantileList | update(QuantileList list)  Updates the existing list of Quantile objects with specified list for a given uuid.  Returns this newly updated List of Quantile objects. |
| QuantileList | remove(byte[] uuid)  Deletes and returns a complete List of Quantile objects associated with specified uuid from database table. |

#### The Power Method Retrieve Resource

The Quantile Retrieve Resource is the external interface for retrieving a list of Quantile objects associated with specified uuid.

URI to call the Quantile Retrieve Server Resource class is

http://tomcat-server/study/quantileList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| QuantileList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Quantile objects associated with the specified uuid from database. |

### The Response List Resources

#### The Responses Resource

The Responses Resource is the external interface for creating/updating/deleting list of Responses objects. ResponsesList object is a domain layer class, for more details please refer section 3.1.44 of Domain Layer Specification Document.

URI to call the Responses Server Resource class is

http://tomcat-server/study/responseList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| ResponsesList | Create(ResponsesList list)  Creates, saves and returns a list of Response objects associated with specified uuid. |
| ResponsesList | update(ResponsesList list)  Updates the existing list of Response objects with specified list for a given uuid.  Returns this newly updated List of Response objects. |
| ResponsesList | remove(byte[] uuid)  Deletes and returns a complete List of Response objects associated with specified uuid from database table. |

#### The Responses Retrieve Resource

The Responses Retrieve Resource is the external interface for retrieving a list of Response objects associated with specified uuid.

URI to call the Responses Retrieve Server Resource class is

http://tomcat-server/study/responseList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| ResponsesList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Response objects associated with the specified uuid from database. |

### The Nominal Power List Resources

#### The Nominal Power Resource

The Nominal Power Resource is the external interface for creating/updating/deleting list of Nominal Power objects. NominalPowerList object is a domain layer class, for more details please refer section 3.1.42 of Domain Layer Specification Document.

URI to call the Nominal Power Server Resource class is

http://tomcat-server/study/nominalPowerList

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| NominalPowerList | Create(NominalPowerList list)  Creates, saves and returns a list of Nominal Power objects associated with specified uuid. |
| NominalPowerList | update(NominalPowerList list)  Updates the existing list of Nominal Power objects with specified list for a given uuid.  Returns this newly updated List of Nominal Power objects. |
| NominalPowerList | remove(byte[] uuid)  Deletes and returns a complete List of Nominal Power objects associated with specified uuid from database table. |

#### The Nominal Power Retrieve Resource

The Nominal Power Retrieve Resource is the external interface for retrieving a list of Response objects associated with specified uuid.

URI to call the Nominal Power Retrieve Server Resource class is

http://tomcat-server/study/nominalPowerList/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| NominalPowerList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Nominal Power objects associated with the specified uuid from database. |

### The Confidence Interval Description Resources

#### The Confidence Interval Resource

The Confidence Interval Resource is the external interface for creating/updating/deleting Confidence Interval object. UuidConfidenceIntervalDescription object is a domain layer class, for more details please refer section 3.1.26 of Domain Layer Specification Document.

URI to call the Confidence Interval Server Resource class is

http://tomcat-server/study/confidenceIntervalDescription

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| UuidConfidenceIntervalDescription | Create(UuidConfidenceIntervalDescription list)  Creates, saves and returns a Confidence Interval Description object associated with specified uuid. |
| UuidConfidenceIntervalDescription | update(UuidConfidenceIntervalDescription list)  Updates the existing Confidence Interval Description object with specified object for a given uuid.  Returns this newly updated Confidence Interval Description object. |
| UuidConfidenceIntervalDescription | remove(byte[] uuid)  Deletes and returns Confidence Interval Description object associated with specified uuid from database table. |

#### The Confidence Interval Retrieve Resource

The Confidence Interval Retrieve Resource is the external interface for retrieving a Confidence Interval Description object associated with specified uuid.

URI to call the Confidence Interval Retrieve Server Resource class is

http://tomcat-server/study/confidenceIntervalDescription/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| UuidConfidenceIntervalDescription | retrieve(byte[] uuid)  Retrieves and returns list of Confidence Interval Description objects associated with the specified uuid from database. |

### The Power Curve Description Resources

#### The Power Curve Resource

The Power Curve Resource is the external interface for creating/updating/deleting Power Curve Description object. UuidPowerCurveDescription object is a domain layer class, for more details please refer section 3.1.24 of Domain Layer Specification Document.

URI to call the Power Curve Server Resource class is

http://tomcat-server/study/powerCurveDescription

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| UuidPowerCurveDescription | Create(UuidPowerCurveDescription list)  Creates, saves and returns a Power Curve Description object associated with specified uuid. |
| UuidPowerCurveDescription | update(UuidPowerCurveDescription list)  Updates the existing Power Curve Description object with specified object for a given uuid.  Returns this newly updated Power Curve Description object. |
| UuidPowerCurveDescription | remove(byte[] uuid)  Deletes and returns Power Curve Description object associated with specified uuid from database table. |

#### The Power Curve Retrieve Resource

The Power Curve Retrieve Resource is the external interface for retrieving a Power Curve Description object associated with specified uuid.

URI to call the Power Curve Retrieve Server Resource class is

http://tomcat-server/study/powerCurveDescription/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| UuidPowerCurveDescription | retrieve(byte[] uuid)  Retrieves and returns list of Power Curve Description objects associated with the specified uuid from database. |

### The Clustering Resources

#### The ClusterNode Resource

The ClusterNode Resource is the external interface for creating/updating/deleting Clustering object. ClusteNodeList object is a domain layer class, for more details please refer section 3.1.12 of Domain Layer Specification Document.

URI to call the Cluster Server Resource class is

http://tomcat-server/study/clustering

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| ClusterNodeList | Create(ClusterNodeList list)  Creates, saves and returns a list of Cluster Node objects associated with specified uuid. |
| ClusterNodeList | update(ClusterNodeList list)  Updates the existing list of Cluster Node objects with specified object for a given uuid.  Returns this newly updated Cluster Node List object. |
| ClusterNodeList | remove(byte[] uuid)  Deletes and returns a complete list of Cluster Node objects associated with specified uuid from database table. |

#### The ClusterNode retrieve Resource

The Clustering Retrieve Resource is the external interface for retrieving a cluster node list object associated with specified uuid.

URI to call the Cluster Retrieve Server Resource class is

http://tomcat-server/study/clustering/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| ClusterNodeList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Cluster Node objects associated with the specified uuid from database. |

### The Between Participant Factor Resources

#### The Between Participant Resource

The Between Participant Resource is the external interface for creating/updating/deleting Between Participant Factors object. BetweenParticipantFactorList object is a domain layer class, for more details please refer section 3.1.10 of Domain Layer Specification Document.

URI to call the Between Participant Server Resource class is

http://tomcat-server/study/betweenParticipantFactor

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| BetweenParticipantFactorList | Create(BetweenParticipantFactorList list)  Creates, saves and returns a list of Between Participant Factors object associated with specified uuid. |
| BetweenParticipantFactorList | update(BetweenParticipantFactorList list)  Updates the existing list of Between Participant Factor object with specified object for a given uuid.  Returns this newly updated Between Participant Factor List object. |
| BetweenParticipantFactorList | remove(byte[] uuid)  Deletes and returns a complete list of Between Participant Factors object associated with specified uuid from database table. |

#### The Between Participant retrieve Resource

The Between Participant Factors Retrieve Resource is the external interface for retrieving a Between Participant Factors list object associated with specified uuid.

URI to call the Between Participant Retrieve Server Resource class is

http://tomcat-server/study/betweenParticipantFactor/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| BetweenParticipantFactorList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Between Participant Factor objects associated with the specified uuid from database. |

### The Repeated Measures Resources

#### The Repeated Measures Resource

The Repeated Measures Resource is the external interface for creating/updating/deleting Repeated Measures Node object. RepeatedMeasuresNodeList object is a domain layer class, for more details please refer section 3.1.14 of Domain Layer Specification Document.

URI to call the Repeated Measures Server Resource class is

http://tomcat-server/study/repeatedMeasures

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| RepeatedMeasuresNodeList | Create(RepeatedMeasuresNodeList list)  Creates, saves and returns a complete list of Repeated Measures Node objects associated with specified uuid. |
| RepeatedMeasuresNodeList | update(RepeatedMeasuresNodeList list)  Updates the existing list of Repeated Measures Node objects with specified object for a given uuid.  Returns this newly updated Repeated Measures Node List object. |
| RepeatedMeasuresNodeList | remove(byte[] uuid)  Deletes and returns a complete list of Repeated Measures Node objects associated with specified uuid from database table. |

#### The Repeated Measures retrieve Resource

The Repeated Measures Retrieve Resource is the external interface for retrieving a Repeated Measures node list object associated with specified uuid.

URI to call the Repeated Measures Retrieve Server Resource class is

http://tomcat-server/study/repeatedMeasures/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| RepeatedMeasuresNodeList | retrieve(byte[] uuid)  Retrieves and returns a complete list of Repeated Measures Node objects associated with the specified uuid from database. |

### The Matrix Resources

#### The Matrix Resource

The Matrix Resource is the external interface for creating/updating/deleting NamedMatrix object. UuidMatrix object is a domain layer class, for more details please refer section 3.1.7 of Domain Layer Specification Document.

URI to call the Matrix Server Resource class is

http://tomcat-server/study/matrix

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| NamedMatrix | Create(UuidMatrix uuidMatrix)  Creates, saves and returns a NamedMatrix object associated with specified uuid. |
| NamedMatrix | update(UuidMatrix uuidMatrix)  Updates the existing NamedMatrix object with specified object for a given uuid.  Returns this newly updated NamedMatrix object. |
| NamedMatrix | remove(UuidMatrixName uuidName)  Deletes and returns a NamedMatrix object associated with specified uuid from database table. |

#### The Matrix retrieve Resource

The Matrix Retrieve Resource is the external interface for retrieving a matrix object associated with specified uuid.

URI to call the Matrix Retrieve Server Resource class is

http://tomcat-server/study/matrix/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| NamedMatrix | retrieve(UuidMatrixName uuidMatrixName)  Retrieves and returns NamedMatrix object associated with the specified uuid from database. |

#### The Matrix Set Resource

The MatrixSet Resource is the external interface for creating/updating/deleting Set of NamedMatrix objects. NamedMatrixSet object is a domain layer class, for more details please refer section 3.1.6 of Domain Layer Specification Document.

URI to call the MatrixSet Server Resource class is

http://tomcat-server/study/matrixSet

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| NamedMatrixSet | Create(NamedMatrixSet matrixSet)  Creates, saves and returns a set of NamedMatrix objects associated with specified uuid. |
| NamedMatrixSet | update(NamedMatrixSet matrixSet)  Updates the existing set of NamedMatrix objects associated with specified object for a given uuid.  Returns this newly updated NamedMatrix object. |
| NamedMatrixSet | remove(byte[] uuid)  Deletes and returns a complete set of NamedMatrix objects associated with specified uuid from database table. |

#### The Matrix Set retrieve Resource

The MatrixSet Retrieve Resource is the external interface for retrieving a matrix object associated with specified uuid.

URI to call the MatrixSet Retrieve Server Resource class is

http://tomcat-server/study/matrixSet/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| NamedMatrixSet | retrieve(byte[] uuid)  Retrieves and returns a complete set of NamedMatrix objects associated with the specified uuid from database. |

### The Covariance Resources

#### The Covariance Resource

The Covariance Resource is the external interface for creating/updating/deleting Covariance object. UuidCovariance object is a domain layer class, for more details please refer section 3.1.21 of Domain Layer Specification Document.

URI to call the Covariance Server Resource class is

http://tomcat-server/study/covariance

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| Covariance | Create(UuidCovariance uuidCovariance)  Creates, saves and returns a Covariance object associated with specified uuid. |
| Covariance | update(UuidCovariance uuidCovariance)  Updates the existing Covariance object with specified object for a given uuid.  Returns this newly updated Covariance object. |
| Covariance | remove(UuidCovarianceName uuidName)  Deletes and returns a Covariance object associated with specified uuid from database table. |

#### The Covariance retrieve Resource

The Covariance Retrieve Resource is the external interface for retrieving a Covariance object associated with specified uuid.

URI to call the Covariance Retrieve Server Resource class is

http://tomcat-server/study/covariance/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| Covariance | retrieve(UuidCovarianceName uuidMatrixName)  Retrieves and returns Covariance object associated with the specified uuid from database. |

#### The Covariance Set Resource

The Covariance Set Resource is the external interface for creating/updating/deleting Set of Covariance objects. CovarianceSet object is a domain layer class, for more details please refer section 3.1.20 of Domain Layer Specification Document.

URI to call the Covariance Set Server Resource class is

http://tomcat-server/study/covarianceSet

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| CovarianceSet | Create(CovarianceSet covarianceSet)  Creates, saves and returns a set of Covariance objects associated with specified uuid. |
| CovarianceSet | update(CovarianceSet covarianceSet)  Updates the existing set of Covariance objects associated with specified object for a given uuid.  Returns this newly updated Covariance object. |
| CovarianceSet | remove(byte[] uuid)  Deletes and returns a complete set of Covariance objects associated with specified uuid from database table. |

#### The Covariance Set retrieve Resource

The Covariance Set Retrieve Resource is the external interface for retrieving a covariance object associated with specified uuid.

URI to call the Covariance Set Retrieve Server Resource class is

http://tomcat-server/study/covarianceSet/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| CovarianceSet | retrieve(byte[] uuid)  Retrieves and returns a complete set of Covariance objects associated with the specified uuid from database. |

### The Hypothesis Resources

#### The Hypothesis Resource

The Hypothesis Resource is the external interface for creating/updating/deleting Hypothesis object. UuidHypothesis object is a domain layer class, for more details please refer section 3.1.17 of Domain Layer Specification Document.

URI to call the Hypothesis Server Resource class is

http://tomcat-server/study/hypothesis

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| Hypothesis | Create(UuidHypothesis uuidHypothesis)  Creates, saves and returns a Hypothesis object associated with specified uuid. |
| Hypothesis | update(UuidHypothesis uuidHypothesis)  Updates the existing Hypothesis object with specified object for a given uuid.  Returns this newly updated Hypothesis object. |
| Hypothesis | remove(UuidHypothesisType uuidType)  Deletes and returns a Hypothesis object associated with specified uuid from database table. |

#### The Hypothesis retrieve Resource

The Hypothesis Retrieve Resource is the external interface for retrieving a Hypothesis object associated with specified uuid.

URI to call the Hypothesis Retrieve Server Resource class is

http://tomcat-server/study/hypothesis/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| Hypothesis | retrieve(UuidHypothesisType uuidHypothesisType)  Retrieves and returns Hypothesis object associated with the specified uuid from database. |

#### The Hypothesis Set Resource

The Hypothesis Set Resource is the external interface for creating/updating/deleting Set of Hypothesis objects. Hypothesis object is a domain layer class, for more details please refer section 3.1.16 of Domain Layer Specification Document.

URI to call the Hypothesis Set Server Resource class is

http://tomcat-server/study/hypothesisSet

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| HypothesisSet | Create(HypothesisSet hypothesis Set)  Creates, saves and returns a set of Hypothesis objects associated with specified uuid. |
| HypothesisSet | update(HypothesisSet hypothesis Set)  Updates the existing set of Hypothesis objects associated with specified object for a given uuid.  Returns this newly updated Hypothesis objects. |
| HypothesisSet | remove(byte[] uuid)  Deletes and returns a complete set of Hypothesis objects associated with specified uuid from database table. |

#### The Hypothesis Set retrieve Resource

The Hypothesis Set Retrieve Resource is the external interface for retrieving a Hypothesis object associated with specified uuid.

URI to call the Hypothesis Set Retrieve Server Resource class is

http://tomcat-server/study/hypothesisSet/retrieve

The following methods are supported

|  |  |
| --- | --- |
| Method Summary | |
| HypothesisSet | retrieve(byte[] uuid)  Retrieves and returns a complete set of Hypothesis objects associated with the specified uuid from database. |

## Automated Database Cleanup

Study Designs older than one month will be removed from the database. A crontab entry will be added which will issue a ‘curl’ command to the Study Design service. The curl command will use the /study/study/date/<iso-date> API.

# Reuse and Relationships to Other Products

## Reuse of existing code

* The Study Design Service code will follow the Restlet format used for existing web services, but reuse of code will be minimal.
* Unit testing artifacts and test cases will reuse some of the existing documentation and described test cases.

## Third Party Dependencies

* MySQL Database Server – 5.5

MySQL Administrator version – 1.2.17

MySQL Work Bench version – 5.2.35

* Tomcat 6.0 or higher.
* Hibernate 4.0.0
* Restlet 2.x

# Design Decisions and Tradeoffs

The proposed architecture is not yet fully refined. The following represent open design questions.

1. Synchronization between the Study Design Service database and user-downloaded copies of the study design

Suppose that a user has created a study design which has been stored in the Study Design Service. The user then downloads a local copy of the study design as an xml file. Suppose the user modifies the study design through the user interface but does not save a new copy of the study design. Now the user wants to upload this data. The UUID of the study exists in the Study Design Service, but it out of sync with the uploaded copy. The application can either

* 1. Inform the user of the conflict through an alert box stating ‘This study already exists in table. Do you want to overwrite it?’ The user can indicate whether to overwrite the existing design.
  2. Always overwrite the existing design in the Study Design Service.

1. Doubt in Hypothesis object : ENUM('Main Effects','Interaction','Linear Trend','Quad Trend','Cubic Trend')

Possible solution for this can be use of passwords. But then some more issues come in picture e.g.

1. What is the secure method for storing Passwords in MySQL tables?
2. User has to remember different passwords for each of the study design.
3. And what if user doesn’t remember Password? How can he recover the Password?

# Appendices

## Diagrams

* ***Existing Structure***

TOMCAT

Client browser

Matrix Service

Apache Httpd

Power Service

Chart Service

File Service

* ***Proposed Version 2.0.0 Structure***

TOMCAT

Httpd

Client browser

Study Design Service

Matrix Service

Power Service

Databases

MYSQL

HIBERNATE

File Service