# Part I

# System and Software Design Description (SSDD): Incorporating Architectural Views and Detailed Design Criteria for Groups in a Univeristy Setting

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Identification

The software system being considered for development is referred to as Groups in a University Setting or gus. The customer providing specifications for the ethnic and religion team is the Lutheran Campus Ministry. The ultimate customer, or end-user, of the system will be student groups at the University of Idaho. This is a new project effort, so the version under development is version 0.05

## 1.2 Purpose

The purpose of the system under development is to provide a tool for the easy administration and tracking of university-style groups including but not limited to clubs and sports teams, the system will also try to increase student involvement by connecting and recognizing the involvement of users. While the system will be used by university personnel, this document is intended to be read and understood by UICS software designers and coders. This document will also be approved by Dr. Clinton Jeffery.

## 1.3 Scope

GUS is to social-networking as an intranet is to the Internet. Where other social networks distract users with non-university non-local non-face-to-face non-involvement, GUS will focus these types of functionalities to the university and local community setting to best meet the administrative and service needs of groups in a university setting. In addition to being a

group-centered student-involvement web-application.

## 1.4 Definitions, Acronyms, and Abbreviations

| Term or Acronym | Definition |
|---|---|
| Alpha test | Limited release(s) to selected, outside testers |
| Beta test | Limited release(s) to cooperating customers wanting early access to developing systems |
| Final test | aka, Acceptance test, release of full functionality to customer for approval |
| DFD | Data Flow Diagram |
| SDD | Software Design Document, aka SDS, Software Design Specification |
| SRS | Software Requirements Specification |
| SSRS | System and Software Requirements Specification |
| GUS | Groups in a University Setting |

## 1.5 References


1. www.churchteams.com
2. www.groupmeister.com
3. www.teamr.com
4. www.salesboom.com
5. www.wikipedia.org


## 1.6 Overview and Restrictions

This document is for limited release only to UI CS personnel working on the project.

Section 2 of this document describes the system under development from a holistic point of view. Functions, characteristics, constraints, assumptions, dependencies, and overall requirements are defined from the system-level perspective.

Section 3 of this document describes the specific requirements of the system being developed. Interfaces, features, and specific requirements are enumerated and described to a degree

sufficient for a knowledgeable designer or coder to begin crafting an architectural solution to the proposed system.

Section 4 provides the requirements traceability information for the project. Each feature of the system is indexed by the SSRS requirement number and linked to its SDD and test references.

Sections 5 and up are appendices including original information and communications used to create this document.

# Chapter 2

# Constraints and Stakeholder Concerns

## 2.1 Constraints

### 2.1.1 Environmental Constraints

### 2.1.2 System Requirement Constraints

### 2.1.3 User Characteristic Constraints

## 2.2 Stakeholder Concerns

# Chapter 3

# System and Software Architechture

## 3.1 Users Architectural View

### 3.1.1 User's View Identification

### 3.1.2 User's View Representation and Description

## 3.2 Developer's Architectural View

### 3.2.1 Developer's View Identification

### 3.2.2 Developer's View Representation and Description

#### Object Model

#### Dynamic Model

### 3.2.3 Developer's Architectural Rationale

## 3.3 Consistency of Architecutural Views

### 3.3.1 Developer's Viewpoint Detailed Software Design

### 3.3.2 Component Dictionary

### 3.3.3 Component Detailed Design

#### Detailed Design for Component/Entity: Name of Component

#### Detailed Design for Component/Entity: Name of Component

#### Detailed Design for Component/Entity: Name of Component

#### Detailed Design for Component/Entity: Name of Component

## 3.4 Data Dictionary

# Chapter 4

# Requirements Traceability

# Part II

# Systems and Software Requirements Specification (SSRS) for Groups in a University Setting

# Chapter 5

# Overall Description

## 5.1 Product Perspective

Gus is an independent software system, as it does not directly integrate with a larger system. However, GUS does draw data from external sources, such as personal information databases, and needs to be integrated with a web server in order to be readily accessible.

## 5.2 Product Functions

1. Simplifying tasks to leaders of groups, such as:
    (a) Sending notifications to group members, prospective members, former members, and interested community members (email)
    (b) Sending information (files) to group members via email or download link
    (c) Managing a group-wide calendar of events
        i. track volunteers, attendees, and contributors
        ii. suggest potentially beneficial services other groups could provide related to the event
        iii. provide a calendar of events that includes events from other groups that members would want to attend (like marching band if half of LCM are in the marching band.)
        iv. keeping track of who is responsible for bringing / doing what at an event
    (d) Automatically generating:

i. Contact information (contact sheets, phone directories)
ii. Website with updated contact, group, event, and customized information
iii. Organization charts
iv. Graphical relationships between groups
v. Fees, dues, and expenses notifications
vi. Event reminders

2. Consolidating information for members, former members, potential members (and parents) of groups:
   (a) Common location of group information
   (b) Searching existing groups
   (c) Tying together existing groups (even suggesting similar groups)
   (d) Personalized emails regarding changes/updates
   (e) Outstanding expenses or reimbursements
   (f) Reliable (i.e., automatically updated):
      i. Group contact information
      ii. Group event information
   (g) Transcript of verified group activity (for use with service-learning classes, and proof of volunteerism for potential employers)
   (h) Supplementing Vandal Friday with emails to prospective high school Seniors
3. Getting member input though: forums, project managers, surveys, and polls
4. Payment processing and sponsorship collection
5. 4. Recruitment and advertising for groups, volunteer / paid opportunities, services provided, possibly a bartering tool

## 5.3 User Characteristics

Gus should be easy for any user to understand with a brief explanation and intuitive enough for an uninitiated user to figure out by looking through the options. Basic computer use skills and a simple conceptual explanation should be enough for every day usage.

## 5.4 Constraints

GUS must meet privacy policies as they apply to both the University of Idaho and social networking sites. GUS must be able to interface with outside database servers (such as the Center for Volunteerism's database, UI's career seeker site, common social networks, and parent groups of university groups). Member's activities and group's activities must be audited for accuracy and safety. The languages used to program GUS will be primarily, HTML, CSS, and PHP for the user interface, C++ for the interface between the user interface (which will implement security and complex business rules), and the database, and SQL for the database. The networking protocols will be TCP/IP and Open MP / MPI will be used to enhance parallel operation. The system will have personal information for over 5,000 students, so confidentiality is of the utmost importance.

## 5.5 Assumptions and Dependencies

The software system should run like a web-app and need not be downloaded by users. It is assumed that users will be running Internet Explorer, Fire Fox, or another popular web browser. The server for the system is expected to run a UNIX operating system.

## 5.6 System Level (Non-Functional) Requirements

### 5.6.1 Site dependencies

GUS will require a server that can support 1,000 concurrent users. The database must store the information, interests, and activities of approximately 5,000 external users, 5,000 students and 200 groups.

### 5.6.2 Safety, security and privacy requirements

GUS contains the personal information of over 5,000 users security should be integrated into every facet of this program. The privacy criteria for this system must reflect privacy policies that apply to the University of Idaho, and the security criteria for this system must reflect the need to secure over 5,000 users from identity theft and potential defamation of character.

### 5.6.3 Performance requirements

1. The number of simultaneous users to be supported are: 1,000.

2. Supported information ranges from text to files to streaming video.

3. 95% of the transactions shall be processed in less than half a second.

### 5.6.4 System and software quality

Gus must perform all required functions, behave consistently and correctly, be easily corrected, running between 5:30 am all day to 1:30 am be easily adaptable, test-driven, and easy to use.

### 5.6.5 Packaging and delivery requirements

The executable system and all associated documentation (i.e., SSRS, SDD, code listing, test plan (data and results), and user manual) will be delivered to the customer via Internet download. The final, edited version of the above documents will accompany the final, accepted version of the executable system.

### 5.6.6 Personnel-related requirements

The system under development will require a graduate student system-level administrator to maintain the system.

### 5.6.7 Training-related requirements

No training materials or expectations are tied to this project other than the limited help screens built into the software and the accompanying user manual.

### 5.6.8 Logistics-related requirements

A server will be required to maintain the software system. The user will be required to have an Internet connection.

### 5.6.9 Precedence and criticality of requirements

1. Maintaining confidentiality and privacy of PII

2. This system must be reliable enough for users to not give up on it

3. All other features are less important than the first two and equally important

# Chapter 6

# Specific Requirements

## 6.1 External Interface Requirements

### 6.1.1 Hardware Interfaces

The system will require a server and secure networking abilities.

### 6.1.2 Software Interfaces

The system will require an interface to interact with emailing systems, databases, and authentication servers.

### 6.1.3 User Interfaces

The system will require user interfaces for non-university users (prospective students, community members, alumni, parent groups, etc.), students, officers, and staff/faculty.

### 6.1.4 Other Communication Interfaces

GUS will interface with the university career seeking site and social networking sites.

Table 6.1: Hardware Interfaces

| Name | Source/Destination | Description | Type/range | Dependencies | Formats |
|---|---|---|---|---|---|
| HTTP Server | Dedicated Server or VPS / Client | This device is responsible for serving HTML content (and other content) to clients. Preferably Apache2. | All | Requires a server-capable machine | N/A |
| VPS or Dedicated Server | NA | A VPS or a Dedicated Server, preferably running a pre-configured Linux distribution such as Fedora or Ubuntu. | All | Electricity, high-speed Internet connection | N/A |

Table 6.2: Software Interfaces

| Name | Source/Destination | Description | Type/range | Dependencies | Formats |
|---|---|---|---|---|---|
| SQL Server | Dedicated Server or VPS / Client | Works in conjunction with HTTP server to provide data. | All | Requires a server-capable machine | N/A |
| PHP5 | HTTP Server / Client | Provides computational power so tasks that serve HTML content via apache can be completed. | Requires a server capable of running PHP5. | Electricity, high-speed Internet connection | N/A |

Table 6.3: User Interfaces

| Name | Source/Destination | Description | Type/range | Dependencies | Formats |
|---|---|---|---|---|---|
| Website | HTTP Server/Client | Allows user to interact with the service | All | HTTP Server | Web |
| Cell phone | Cellphone | Receive text-messages | All | HTTP Server | Text |

Table 6.4: Other Communication Interfaces

| Name | Source/Destination | Description | Type/range | Dependencies | Formats |
|---|---|---|---|---|---|
| | | | | | |

## 6.2 System Features

### 6.2.1 Use Case Diagrams

### 6.2.2 Use Case Diagrams

insert 1+ use case diagrams here

### 6.2.3 System feature 1: Officer Management (Administration Module)

**Add Officer**

- **Actors:** Group Administrator
- **Goals:** Administrator will add an officer to the group
- **Preconditions:** Administrator is logged onto to the group's administration page
- **Summary:** Related use cases: modify officer access
- **Steps:**
    1. Administrator clicks add officer button
    2. Administrator types in officer's username
    3. Administrator selects the officer's title/role
    4. Administrator modifies officer's access
    5. Administrator selects save and exit
- **Alternatives:** Administrator cancels out of use case
- **Postconditions:** the selected user is given officer status in the group

**Modify Officer Access**

- **Actors:** Group Administrator
- **Goals:** Administrator will change the access given to an officer
- **Preconditions:** Administrator is logged onto to the group's administration page

- **Summary:** Related use cases:
- **Steps:**
    1. Administrator clicks modify officer access button
    2. Administrator selects officer / role to be modified
    3. Administrator selects type of access to be modified
    4. Administrator selects save and exit
- **Alternatives:** Administrator cancels out of use case
- **Postconditions:** The selected officer(s) access is modified

**Remove Officer**

- **Actors:** Group Administrator
- **Goals:** Administrator will remove an officer from the group
- **Preconditions:** Administrator is logged onto to the group's administration page
- **Summary:** Related use cases:
- **Steps:**
    1. Administrator clicks remove officer button
    2. Administrator selects officer
    3. Administrator selects save and exit
- **Alternatives:** Administrator cancels out of use case
- **Postconditions:** the selected user is removed as a group officer

**Change Group Password**

- **Actors:** Group Administrator
- **Goals:** Administrator will change the password to access the group page
- **Preconditions:** Administrator is logged onto to the group's administration page

- **Summary:** Related use cases:
- **Steps:**
    1. Administrator clicks change group password button
    2. Administrator types in the new password twice
    3. Administrator selects save and exit
- **Alternatives:** Administrator cancels out of use case
- **Postconditions:** the group password is changed

**Edit Forum Priveleges**

- **Actors:** Group Administrator
- **Goals:** Administrator will select who can make changes to forums
- **Preconditions:** Administrator is logged onto to the group's administration page
- **Summary:** Related use cases:
- **Steps:**
    1. Administrator clicks forum priveleges button
    2. Administrator selects who's priveleges will be changed
    3. Administrator selects the the priveleges
    4. Administrator saves and exits
- **Alternatives:** Administrator cancels out of use case
- **Postconditions:** the user's priveledges are changed

### 6.2.4 System feature 2: Group Management (Officer Module)

**Add Member**

- **Actors:** Officer
- **Goals:** Officer will add a member to the group

- **Preconditions:** Officer is logged onto to the group's officer page
- **Summary:** Related use cases:
- **Steps:**
    1. Officer clicks add member button
    2. Officer types in the member's username
    3. Officer selects save and exit
- **Alternatives:** Officer cancels out of use case
- **Postconditions:** the selected user is given membership status in the group

**Remove Member**

- **Actors:** Officer
- **Goals:** Officer will remove member from the group
- **Preconditions:** Officer is logged onto to the group's officer page
- **Summary:** Related use cases:
- **Steps:**
    1. Officer clicks remove member button
    2. Officer selects member
    3. Officer selects save and exit
- **Alternatives:** Officer cancels out of use case
- **Postconditions:** the selected user is removed from the group

**Add Event**

- **Actors:** Officer
- **Goals:** Add an event to fit officer specifications
- **Preconditions:** Officer is logged onto to the group's officer page

- **Summary:** Related use cases: Modify Event
- **Steps:**
    1. Officer clicks Events button
    2. Officer clicks Add New Event
    3. Officer adds a title
    4. Officer chooses a date, time, and recurrence
    5. Officer checks the post to site option
    6. Officer checks the email notifications to members option
    7. Officer modifies the message to be sent
    8. Officer selects save and exit
- **Alternatives:** Officer cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** Event has been added to fit the officer's specifications

**Modify Event**

- **Actors:** Officer
- **Goals:** modify (add, edit, remove) an event on the calendar
- **Preconditions:** Officer is logged onto to the group's officer page
- **Summary:** Related use cases: Add Event, Remove Event
- **Steps:**
    1. Officer clicks Events button
    2. Officer clicks Modify Event button
    3. Officer modifies event information
    4. Officer sends notfications and has the site updated as needed
    5. Officer selects save and exit
- **Alternatives:** Officer cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** Event has been modified to fit the officer's specifications

**Remove Event**

- **Actors:** Officer
- **Goals:** remove an event on the calendar
- **Preconditions:** Officer is logged onto to the group's officer page
- **Summary:** Related use cases: Add Event, modify
- **Steps:**
    1. Officer clicks Events button
    2. Officer clicks Remove Event button
    3. Officer sends notfications and has the site updated as needed
    4. Officer selects save and exit
- **Alternatives:** Officer cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** Event has been removed from calendar and site

**Send Group Email**

- **Actors:** Officer
- **Goals:** the officer will be able to send email to the group
- **Preconditions:** Officer is logged onto to the group's email page
- **Summary:** Related use cases: Add Event, modify event
- **Steps:**
    1. Officer clicks the compose email button
    2. Officer composes an email
    3. Officer selects which subgroup the email goes to
    4. Officer clicks the send button
- **Alternatives:** Officer cancels out of use case by clicking on a different navigational area or clicking Cancel

- **Postconditions:** email has been spellchecked, and sent. Officer is returned to the group's email page

**Modify Website**

- **Actors:** Officer
- **Goals:** to generate a web site of information about a group
- **Preconditions:** Officer is logged onto to the group's officer page, website is already posted and linked to GUS
- **Summary:** Related use cases:
- **Steps:**
  1. Click Web Site button
  2. Click Web site options
  3. Modify information on website
  4. Click Preview button
  5. Select save and exit
- **Alternatives:** Officer cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** the website is published on the website, and automatically linked to from its supergroup

### 6.2.5 System feature 3: User Menu

**Edit Email**

- **Actors:** User
- **Goals:** user will be able to send emails to other GUS users
- **Preconditions:** user is logged onto GUS
- **Summary:** Related use cases:
- **Steps:**

1. User clicks Email button
2. User clicks on the Drafts folder
3. User clicks on the email to be edited
4. User edits email
5. User sends email, and is returned to the drafts page

- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** Email is sent and user is returned to drafts page

**Edit Major**

- **Actors:** User
- **Goals:** user will be able to change personal attributes
- **Preconditions:** user is logged onto GUS
- **Summary:** Related use cases:
- **Steps:**

1. User clicks MyProfile button
2. User makes changes to personal fields
3. User selects save and exit

- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** User's profile and personal page is updated, and user is returned to their home page

**Edit Personal Page**

- **Actors:** User
- **Goals:** user will be able to change appearance and content of personal page
- **Preconditions:** user is logged onto GUS

- **Summary:** Related use cases: Change Major
- **Steps:**
    1. User clicks MyPage button
    2. User edits personal information
    3. User selects privacy settings; to include which user's and groups they want seeing their information
    4. User selects what color theme / appearance they want for their page
    5. User selects what items and information they want on their page
    6. User previews their page and clicks save
- **Alternatives:** User can move through page changes non-sequenctially or cancel out
- **Postconditions:** User's personal page is updated

**Join Group**

- **Actors:** User
- **Goals:** user will be able to request to join group
- **Preconditions:** user is logged onto GUS
- **Summary:** Related use cases:
- **Steps:**
    1. User enters group name
    2. User clicks on the join button
    3. User identifies which subgroups they want to join
    4. User selects save and exit
- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** user's request is auto approved, or request sent to group, user is added to group (if approved), and user is returned to the group's page

**Leave Group**

- **Actors:** User
- **Goals:** user will be able to leave group
- **Preconditions:** user is logged onto GUS
- **Summary:** Related use cases:
- **Steps:**
    1. User clicks MyGroups button
    2. User selects the group to drop
    3. User clicks the drop button
- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** User is dropped from the group, the group is removed from the user page and the user is returned to the MyGroups page

**Send Email**

- **Actors:** User
- **Goals:** user will be able to send emails to other GUS users
- **Preconditions:** user is logged onto GUS
- **Summary:** Related use cases:
- **Steps:**
    1. User clicks Email button
    2. User clicks on the Drafts folder
    3. User clicks on the email to be edited
    4. User edits email
    5. User sends email, and is returned to the drafts page

- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** Email is sent and user is returned to drafts page

**Send Text**

- **Actors:** User
- **Goals:** user will be able to send texts to other GUS users
- **Preconditions:** user is logged onto GUS
- **Summary:** Related use cases:
- **Steps:**
    1. User clicks text button
    2. User types in the recipient
    3. User types the message
    4. User clicks send
    5. User makes any required changes, recipient, spelling (if required) go to step 4
- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** text is sent and user is returned to previous page

**Join Event**

- **Actors:** User
- **Goals:** user will be able to sign up for event
- **Preconditions:** user is logged onto GUS, and at the group's site
- **Summary:** Related use cases:
- **Steps:**
    1. User hovers over the calendar, the system displays the user's calendar next to the group calendar

2. User hovers over the event to join, the system displays event details
3. User right-clicks on the event, and selects will-be-there, or may-be-there
4. User may enter a message

- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** User is added to the event's guest list, the event is added to the user's calendar, and the user is returned to the group page

**Leave Event**

- **Actors:** User
- **Goals:** user will be able to drop an event
- **Preconditions:** user is logged onto GUS
- **Summary:** Related use cases:
- **Steps:**

1. User clicks on MyCalendar
2. User right-clicks on the event
3. User selects drop

- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** User is removed from the event's guest list, the event is removed from the user's calendar, and the user is returned to the calendar page

**Sign Up**

- **Actors:** User
- **Goals:** user will be able to sign up for GUS
- **Preconditions:** user is at GUS page
- **Summary:** Related use cases: Modify personal information, modify personal page

- **Steps:**
    1. User clicks on Sign Up
    2. User completes steps for modifying their page
    3. User selects groups to join
- **Alternatives:** User cancels out of use case by clicking on a different navigational area or clicking Cancel
- **Postconditions:** User is added to GUS, and relevant users/groups are notified

### 6.2.6 System feature 4: Forum Module)

**Add / Post Topic**

- **Actors:** User
- **Goals:** User will add a post or start a topic
- **Preconditions:** User is logged in, and at the group's forum page
- **Summary:** Related use cases:
- **Steps:**
    1. User clicks on topic to post to, or clicks AddTopic button
    2. User completes a post (the default title is the first line of the post)
    3. User clicks post button The system spell-checks the post, and truncates it if it's too long, it also validates the post, and returns an error message (if needed)
    4. If needed user makes corrections and repeats step 3
- **Alternatives:** user cancels out of use case
- **Postconditions:** the post is sent to the group for validation, and the user is returned to the topic page

**Remove / Post Topic**

- **Actors:** User
- **Goals:** User will remove a post or topic
- **Preconditions:** User is logged in, and at the group's forum page
- **Summary:** Related use cases:
- **Steps:**
    1. User clicks on topic or the post to remove
    2. User clicks remove button
- **Alternatives:** User cancels out of use case
- **Postconditions:** If user has the correct permissions, the post or topic is removed, otherwise it is not

**Edit / Post Topic**

- **Actors:** User
- **Goals:** User will edit a post or a topic
- **Preconditions:** User is logged in, and at the group's forum page
- **Summary:** Related use cases: add post/topic
- **Steps:**
    1. User right-clicks on topic or post to be edited
    2. User selects edit
    3. User modifies the post (the default title is the first line of the post)
    4. User clicks post button, the system spell-checks the post, and truncates it if it's too long, it also validates the post, and returns an error message (if needed)
    5. If needed user makes corrections and repeats step 3
- **Alternatives:** user cancels out of use case
- **Postconditions:** the post is sent to the group for validation, and the user is returned to the topic page

# Chapter 7

# Requirements Traceability

## 7.1 ...

### 7.1.1 ...

### 7.1.2 ...

### 7.1.3 ...

### 7.1.4 ...

### 7.1.5 ...

### 7.1.6 ...

## 7.2 ...

### 7.2.1 ...

### 7.2.2 ...

### 7.2.3 ...

### 7.2.4 ...

### 7.2.5 ...

### 7.2.6 ...

## 7.3 ...



### 7.3.1 ...

### 7.3.2 ...

### 7.3.3 ...

# Part III

...

# Appendix A

...

# Appendix B

...