Analysis Report

Table of Contents

Requirement Analysis Importance

Requirements will have to be analyzed after elicitation. There are a number of qualifications that each requirement must possess and every requirement in the set will have to display in order for this group of requirements to be considered valid by developers. The requirements are valid if the requirement construct is valid, characteristics of individual requirements are met, correct language is used, and the right requirement attributes are displayed. All of the following factors are what come together to make a valid requirement. Valid requirements keep the developers on the right track to creating the right software. Analyzing requirements can be very important in preventing future problems within the design process. Analyzing requirements will: resolve conflicts between requirements, discover the bounds of software as well as the interaction between the organizational/ operational environment, and will allow you to derive new software requirements off of system requirements. Careful requirement analysis allows the results of the requirement to be optimal for the product which will be developed for the client minimizing costs.

How Requirements are Analyzed

Requirements are analyzed through a number of processes which will allow the requirement to be validated, implementation to be verified, and the costs to be estimated. Through the process we go through requirements classification, conceptual modeling, architectural design, requirement allocation, requirement negotiation, and a formal analysis of the requirements.

Requirement Classification

In the process of analyzing a requirement the process of classifying the requirement will be made. Classifying the requirement allows the developers to easily know and categorize each requirement so that these requirements can be grouped and followed at separate stages as well as on different levels. Classification involves:

- Whether the requirement is functional or non-functional.
- Whether the requirement is derived or more high-level requirements on emergency property.

- Whether the requirement is on the product or the process.
- The requirement priority.
- The scope of each requirement.
- Volatility/stability (some requirements will change or be removed from the cycle).

Conceptual Modeling

The process of analyzing a requirement has to include conceptual modeling, the development of models of a real-world problem is key to software requirements analysis. The use of the model will aid in understanding of analyzing a solution. Several models can be developed to analyze a software solution. A common software used to create modeling notations is created through using the Unified Modeling Language (UML). The modeling notation to conceptualize a software problem and requirement can depend on:

- The nature of the problem
- The expertise of the software engineer
- The process requirements of the customer

It is essential in almost all cases to build a model of the surrounding software context. The connection can then be made to the surrounding environment of the software and software requirements in question.

Architectural Design and Requirements Allocation

The solution architecture will be derived at some point of the process. Architectural design is the point to which the requirements process overlaps with software and illustrated the impossibility of decoupling the two tasks. The process of analyzing and elaborating the requirements demands that the architecture components must be satisfied so the requirements can be identified. Allocation is essential to permit detailed analysis of the requirements. Once a set of requirements has been allocated to a component, the individual requirements can be further analyzed to discover further requirements on how the component needs to interact with other components in order to satisfy the allocated requirements.

Requirements Negotiation

Conflict resolution is another term for requirement negotiation. Requirement negotiation is when two stakeholders' conflict due to mutually incompatible features between requirements and resources, or possibly between functional and non-functional requirements. The best thing to do in a conflict between stakeholders is to consult between stakeholders to reach a consensus. This is a topic of requirement analysis, but a strong case can be made to push this problem to requirement validation.

It can be also important to make requirement prioritizations which is a necessary analysis process for requirements. Requirement prioritization must occur frequently and be checked consistently so developers have a list of requirements that are organized for most important to least importance level. Use a 1-5 list.

Attributes of a Valid Requirement

Requirements Construct

Stakeholder, system, and system element requirements shall be developed which will help in the future to see if the project captures stakeholder needs. According to IEEE 29148 page 10, it describes that a well-formed specific requirement shall contain one or more of “

- it shall be met or possessed by a system to solve a problem, achieve an objective or address a stakeholder concern;
- it is qualified by measurable conditions;
- it is bounded by constraints;
- it defines the performance of the system when used by a specific stakeholder or the corresponding capability of the system, but not a capability of the user, operator or other stakeholder; and
- it can be verified (e.g., the realization of the requirement in the system can be demonstrated)”.

A requirement shall state the subject of the requirement, what shall be done or a constraint.

Characteristics of individual requirements

Each requirement needs to have some characteristics to make it specific enough for the software to not lead to replicates. The specific requirements according to IEEE xplore include necessary, appropriate, unambiguous, complete, singular, feasible, verifiable, correct, and conforming.

-Necessary- Requirement defines an essential capability, constraint, quality factor, or characteristic.

-Appropriate- Specific intent and amount of detail of the requirement is appropriate to the level of the entity to which it refers.

-Unambiguous- The requirement is stated in such a way that can only be interpreted one way

-Complete- Describes the requirement to meet the need without needing other information to understand.

-Singular- States a single capability, characteristic, constraint, or quality factor.

-Feasible- The requirement can be done with acceptable risk.

-Verifiable-Requirement is structured and worded such that its realization can be proven to the customer' satisfaction at the level of the requirements exists. Verifiability is enhanced when the requirement is measurable.

-Correct. The requirement is an accurate representative of the entity need from which it was transformed.

-Conforming- The individual items conform to an approved standard template and style for writing requirements, when applicable.

Requirements language criteria

A good requirement shall state “what” is needed instead of “how”. Vague and general terms shall be avoided. Ambiguous and vague terms are very hard to identify. Ambiguous terms could include the following: “best”, “most”, “user-friendly”, “cost-effective”, “it”, “that”, “almost always”, “significant”, “minimal”, ambiguous logic statements such as “or” and “and/or” (consider multiple requirements if statement is used), open ended terms such as “provide support”, “higher quality”, loopholes such as “as appropriate”, and terms that apply totality such as “never”/”every”/”all”.

Requirements attributes

To support requirement analysis well-formed requirements are going to include descriptive attributes defined to assist identifying the right information. Attribute information will be associated with the requirements in the requirements repository.

Examples of requirement attributes:

- Identification- Each requirement shall be uniquely identified, identification can reflect linkages and relationships. Unique identifiers will aid requirement tracing.
- Version number- Make sure that the correct version of the requirement is being implemented as well as to provide indication of volatility of the requirement.
- Owner- The person of the organization that maintains the requirement who has the right to say something about this requirement, approves changes, reports status.
- Stakeholder priority- Priority of each requirement shall be identified. A number system can be used to indicate the priority of the requirement like 1-5, 5 being the highest priority.
- Risk- A risk value assigned to each requirement based on risk factors. Include risk to emphasize that the requirement may change or evolve.
- Rationale- Reason that requirement is needed and points to any supporting analysis, trade study, modeling, simulation or other substantiative objective evidence.
- Difficulty- Assumed difficulty for each requirement shall be noted.

- Type- Requirements vary in intent and the kinds of properties they represent. Use a type of attribute aids in identifying relevant requirements and categorizing requirements into groups for analysis and allocation.

Sample Requirement Statement

Requirement Statement:

- When contacting a student, the application shall send one email to the student's associated FGCU email.

Requirement Analysis for FGCU Complete

Within the requirement elicitation process for FGCU Complete, various methods were used to elicit the requirements for the FGCU Complete Emailing Tool. To begin with, an interview was conducted with the primary stakeholder and owner of FGCU Complete which overviewed: business objectives, business operations, as well as the current problems involved within FGCU Complete operations. After the interview was conducted the following practices were applied when analyzing the requirements elicited:

Key	Summary	Labels	P
BAN-47	REQ-20. The program shall include an "Exit" button which safely exits the program and logs out of the system.	Functional	=
BAN-46	REQ-19. The program shall include a "Deselect All" button which deselects every student from email list.	Functional	^
BAN-45	REQ-18. The program shall output a dialog box labeled "Connection Lost" when connection to internet is lost followed by logging out of the system.	Functional	^

BAN-44	REQ-17. The program shall have a "Remove All" button which will clear the list labeled "Selected Students".	Functional	^
BAN-43	REQ-16. The program shall have a "Remove Student" button which will remove student from list.	Functional	^
BAN-42	REQ-15. The program shall have a "Confirm" button which takes the list of students in "select Students" and moves them to "Selected Students"	Functional	^
BAN-41	REQ- 14. The program shall have an "Action" button which will send content from the "Send Email" button to the selected students' emails.	Functional	^
BAN-40	REQ-13. The program shall have a "Save" button which will commit the "Create Email" textbox to the "Send Email" textbox.	Functional	^
BAN-39	REQ-12. The program shall have a "Sign In" button which will log the user in to outlook.	Functional	^
BAN-31	REQ-11. Operators shall be able to change the email the program sends emails from	Functional	⌄⌄
BAN-29	REQ-10. Students from the email list shall receive an email as an output of the program.	Functional	^^
BAN-26	REQ-9, The program shall be robust to all inputs possible from users without any crashes.	Nonfunctional	^^
BAN-25	REQ-8. The program shall only allow the operator to use Florida Gulf Coast University emails	Nonfunctional	⌄
BAN-24	REQ-7. The program shall include a "Send Email" button which sends an email to all students within the email list.	Functional	^
BAN-22	REQ-6. The program shall include a "Deselect" button which deselects a single student from the email list.	Functional	=

BAN-21	REQ-5. The program shall include a "Select" button which selects a single student without any constraints to add to an email list.	Functional	=
BAN-20	REQ-4. The program shall be created using Python	Functional	^
BAN-19	REQ-3. The program shall have a "Select All" button which selects students in constraints of FGCU Complete program to select emails from these students.	Functional	^
BAN-9	REQ-2. The operator shall be able to compose an email through a large textbox with a save button.	Functional	=
BAN-8	REQ-1. The system shall have emailing capacity.	Functional	^^

20 issues

FGCU Complete Requirement Classification

In the above set of requirements, the classification process was performed by adding attribute types to the requirement such as functional or non-functional, requirement number, and requirement priority (scaled from lowest-highest). These classifications can be used to sort the requirements and provide more information to developers of the requirement they will be implementing. Additionally, some requirements will be classified based on the association to the business requirement. The following table provides a sorted list based on classifications and linked to the business requirements. The following applies classification.

BR-ID	Business Requirement	Functional Requirement	Test Cases	Priority
1	Faculty Login	BAN-39 - Getting issue details... STATUS	001- Successful Login scenario with “Sign In”	High

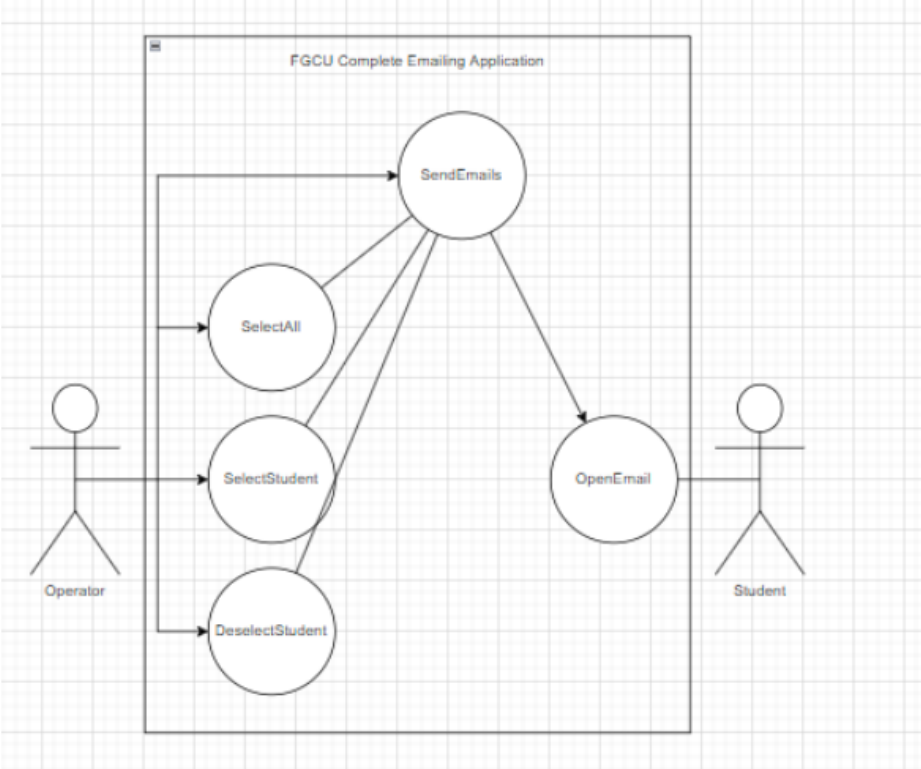
		BAN-25 - Getting issue details... STATUS	002- Successful Login scenario with a faculty-email	Low
2	Email Composition	BAN-40 - Getting issue details... STATUS	003- Successfully clears “Create Email” textbox and transfers contents to “Send Email”.	High
		BAN-31 - Getting issue details... STATUS	004- Successfully can login through separate faculty accounts.	Lowest
		BAN-9 - Getting issue details... STATUS	005- Successful textbox change within “Create Email” textbox.	Highest
3	Select Student	BAN-19 - Getting issue details... STATUS	006- Successfully highlights every student listed from FGCU database.	High
		BAN-21 - Getting issue details... STATUS	007- Successfully highlights single student listed from FGCU database.	Medium
4	Deselect Student	BAN-22 - Getting issue details... STATUS	008- Successfully unselects single student listed from FGCU database.	Medium
		BAN-46 - Getting issue details... STATUS	009- Successfully unselects every student selected from FGCU database.	High
5	Remove Student	BAN-43 - Getting issue details... STATUS	010- Successfully remove student from “Selected Students” list box.	High

		BAN-44 - Getting issue details... STATUS	011- Successfully remove all students from “Selected Students” list box.	High
6	Send Email	BAN-24 - Getting issue details... STATUS	012- Successfully outputs email from “Send Email” textbox to	High
7	Exit Program	BAN-47 - Getting issue details... STATUS	013- Successfully goes to login screen before exiting program	Medium

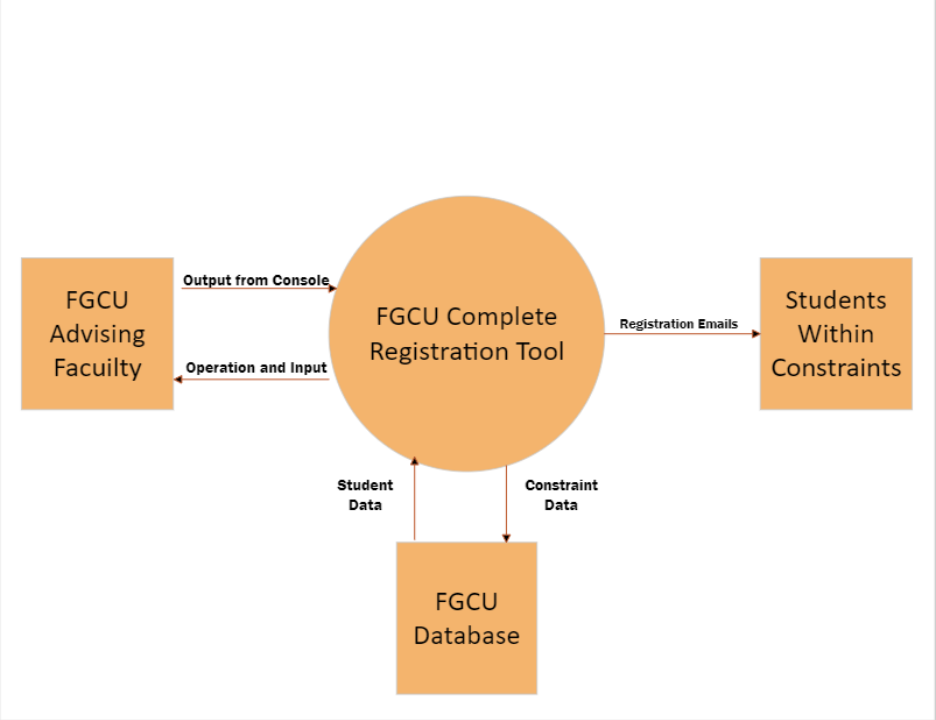
FGCU Complete Conceptual Modeling

Understanding the problem is a key step to creating requirements for a viable software solution. During the process of requirement creation and requirement analysis for a student emailing system, different models were created to further understand the problem advising was facing and where the software was located. This first model of how FGCU Complete emailing would work was created through a use case diagram which utilized the Unified Modeling Language (UML) software http://draw.io . This use case diagram of FGCU Complete features different actors connected to use cases to display the software in creation. Each use-case of the software would illustrate a circle, the circle is connected with a different action of the system the actor can control. The actors in the system involves the operator which is an FGCU faculty member while the second actor is a student with a FGCU associated email.

FGCU COMPLETE



Part of analyzing our software at different levels is by utilizing the correct diagrams to display this information. To provide context to the software it was an important step to label the context of the software and the interactions between the software within its surrounding environment. The context diagram was created using Microsoft Visio to display outside inputs to our FGCU Complete Registration Tool which include the “FGCU Advising Faculty”, the “FGCU Database”, and the output involves registration emails directed to “Students Within Constraints”. The tool can be utilized for analyzing our requirements.



FGCU Complete Requirement Allocation

Within the set of requirements made for the FGCU Complete tool a priority attribute was assigned to each requirement. This attribute is used to describe the allocation that should be given for each requirement in the system by level of importance on a scale of 1-5. Requirement allocation can be utilized by the software team in order to give the team insight to the amount of resources to go into each requirement. The stakeholders value every requirement separately and a priority list is a great tool to give developers insight.

FGCU Complete Elder Paul Critical Thinking Model

Intellectual Standards

The intellectual standard I used during the requirement process involved diving into the intellectual standards of significance, precision, clarity, and relevance. Each requirement was created through the analysis of the significance of the impact each requirement would have on the final software, this was performed by giving each requirement a value of importance on a

scale of 1-5. Within the requirement analysis and creation process, each requirement requires precise standards of declaration following the criteria for language as well as precision. The requirements within the requirement set involves precisely defined requirements, each requirement which can be compounded would be divided into smaller more precise requirements. Concise language to define each function throughout the program is written in the requirement by following the requirement language criteria. Precision within requirement analysis is an important part of the process. Each requirement should be precise to the reader. Precision is used to strictly define each requirement to a specified command for the developer. Precision impacts the level to which the developer is able to build the software, precision is an important factor in the creation of a set of requirements or analyzing a preexisting requirement set. The final important attribute in creating requirements for software development involves relevance. Each requirement includes relevance to a specific portion of the software to be developed. When analyzing requirements each requirement in the requirement set must uniquely identify a separate and relevant feature of an application.

Elements of Reasoning

Within the elements of reasoning, which we apply during the interpretation process of the given software, it is critically important to realize how each intellectual standard of our analyzed set of requirements are interpreted by developers who read these requirements. During the analyzation of requirements for FGCU Complete, an understanding of assumptions, point of view, and concepts was crucial to adjusting the requirement set to be optimally useful for development. For development of software, the assumptions must be correct in order for stakeholders to be represented optimally during development. To create the right assumptions of the system, stakeholders are there to check the requirements frequently which provides the right meaning for the requirement set. The stakeholder was contacted and provided multiple models to obtain the point of view of the client to involve obtaining vision to how the software should look. The concept was illustrated in various ways through the diagrams listed above which was brought back to the stakeholder. During this process, requirements can be refined to meet exactly the stakeholder’s expectations.

Intellectual Traits

The intellectual traits exhibited by the requirement analysis process pertaining to the formulation of the FGCU Complete requirement set involved confidence in reason, intellectual humility, and fairmindedness. The first principle in the analysis of the requirement set involves confidence in reason. The reasoning process in the creation of this requirement set requires a confidence in reason during the analysis of requirements. The set of requirements have justified reasoning through the notes and is consistent within the analysis process. The principle of confidence in reasoning creates a great set of requirements. The following rule that followed the intellectual humility trait was the consistency of the FGCU Complete requirement set. Each requirement is validated and consistently connected to every surrounding requirement in the set. The following two traits also can be applied directly to the fairmindedness used in the requirement set. All requirements are validated through proper analysis and reasoning. The requirement set follows the intellectual traits of confidence in reason, intellectual humility, and fairmindedness.

Software Engineering Ethics Code

To recognize the ethical and professional software responsibilities of the developer it is first important to be aware of the global, economic, environmental, and societal contexts of the software created. In the creation of the following software guidance was followed from the defined list of ethical constructs associated with both The Software Engineering Code of Ethics and Professional Practice published by the ACM, and the NSPE Code of Ethics for Engineers. The following table contains a condensed list of these principals.

1.Public	Software engineers shall act consistently with the public interest.
2. Client and Employer	Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.
3. Product	Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
4. Judgement	Software engineers shall maintain integrity and independence in their professional judgement.

5. Management	Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
6. Profession	Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
7. Colleagues	Software engineers shall be fair to and supportive of their colleagues.
8. Self	Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

The Software Engineering Code of Ethics and Professional Practice (ACM full):

https://ethics.acm.org/code-of-ethics/software-engineering-code/

NSPE Code of Ethics for Engineers:

https://www.nspe.org/resources/ethics/code-ethics