**Version History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Ver. No.*** | ***Authors*** | ***Date*** | ***Reviewers*** | ***Review Date*** | ***Release Date*** |
| 1.0 | Application Development Team | 27-Aug-2018 | QMF | 31-Aug-2018 | 03-Sep-2018 |
| 2.0 | Application Development Team | 10-Dec-2019 | QMF | 13-Dec-2019 | 16-Dec-2019 |
| 3.0 | Application Development Team | 2-Nov-2020 | QMF | 6-Nov-2020 | 10-Nov-2020 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Change History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Ver. No.*** | ***Section*** | ***Date*** | ***Change Information*** | ***RFC No.*** |
| 1.0 | All | 03-Sep-2018 | New Release | - |
| 2.0 | All | 16-Dec-2019 | New Release | - |
| 3.0 | All | 10-Nov-2020 | Annual Review | - |
|  |  |  |  |  |
|  |  |  |  |  |

**GUIDELINES FOR HIGH LEVEL DESIGN**

1. **Objectives**

The objective of this document is to describe the recommended practices for creating the High Level Design (HLD).

1. **Scope**

These guidelines are applicable to all software engineering projects.

1. **Document Structure, Glossary of Terms, etc.**
2. System Architecture
3. Logical Data Model
4. User Procedures
5. User Interface
6. Prototyping
7. Error Messaging Concepts
8. **System Architecture**
9. Identify the entities
10. Draw the Context Level Diagram
11. Translate the functions of BRD into process
12. Prepare short description of the components (entities and processes )
13. Group the process into modules
14. Identify the inputs, outputs and interfaces between the modules
15. Identify the features of the system in terms of :
16. Performance requirements (In case of Web based projects identify:

* The processing to be carried out in the business layer
* The processing to be carried out in the Web pages

1. For optimization of performance, major processing should be carried out in the business layer keeping the processing in the web pages to bare minimum like:
2. User interface requirements
3. Distributed processing
4. Localization (Local language support)
5. Compatibility
6. Audit trails
7. Error log
8. Archiving and restoring
9. Define the security features by considering the following :
10. Name the specific security system to be used
11. Evaluate the feasibility of the specified security requirements with the selected security system
12. Define the default security characteristics (e.g., all allowed, nothing allowed)
13. Define subjects (e.g., user, group, and program)
14. Define authorization hierarchies (e.g., user, group, default)
15. List security-critical items (e.g., data fields, functions)
16. Define authorizations (e.g., grantor = owner of an object, grantee = owner of an authorization for an object)
17. Describe the security logging concept
18. Document the design of the primary security modules
19. Document the design of the authorization validation for client and server components
20. Define error handling concepts by considering the following :
21. Describe the mechanism for displaying an error message to the user
22. Define the different error levels (information, warning, error, fatal error)
23. Describe how to display error messages and return codes from external (callable) systems
24. Where the error texts be will stored (plain ASCII text file, database, etc.)?
25. When will logs be written?
26. Document the design of the primary error-processing modules
27. For web based projects, focus on the business layer and try to minimize processing within web pages
28. **Logical Data Model**
29. Identify the data requirements of the system
30. Prepare a data dictionary
31. Identify entity set
32. Define relationships between these entity sets
33. Define cardinality (one to one, one to many, many to many) from both sides of a relationship between two entities
34. Add attributes to the entities, using the data dictionary elements under each entity
35. Choose identifiers for all entities
36. Perform 3rd level normalization
37. Produce an Entity Life Cycle History diagram (optional - will help define missed transactions in the DFD)
38. Suggested Tools: S-Designer - Data Architect, Erwin, and LBMS
39. Balance the Data Model Diagram with the DFD using the following guidelines
40. Check whether each external entity in a DFD has a corresponding Entity set in the Data Model
41. Check whether each Data store in a DFD has a corresponding Entity set in the Data Model
42. Check the components (data store, external entity) connecting each process and determine if there exists a relationship exists between the components, and if so whether these relationships are defined in the Data model
43. Check if any data flow carries information relating to one entity and determine if these entities are defined in the Data Model
44. **User Procedures**
45. Identify the way the system is expected to be used by the end user in normal day-to-day business and also in abnormal circumstances like data recovery etc.
46. Identify the menus, screens and reports
47. Identify the data requirements for the screens and reports
48. Verify the data requirement with the Logical database design
49. Define the navigation path of the system
50. Define the roles and responsibilities of departments / designations / roles / individuals to install and operate the system
51. **User Interface**

The contents or items to be addressed in a user interface definition needs to specify the following based on the type and requirements of the system:

1. Type of screens and its Navigation (Application Specific)
2. Screen background and foreground color
3. Screen font type, size and color
4. Screen Title Bar Color, Fonts etc.
5. Reports
6. Header standard format (Heading, Date, Time, Page No. etc.)
7. Footer standard format, if any
8. Margins (Left, Right, Top, and Bottom)
9. Type and size of fonts
10. Paper size
11. Layout (landscape / portrait)
12. GUI Standards
13. Tool Bars (Position and Icons)
14. Minimize / Maximize Icons
15. Control menu
16. Window title
17. Date and Currency formats
18. Screen Navigation
19. Menu Hierarchy and Navigation
20. Online messages
21. Type of messages (Error, Warning, Information etc.)
22. Display mechanism (Icons, Windows, and Numbering etc.)
23. Hot keys
24. Help features
25. **Prototyping**
26. Project Manager in conjunction with the Designer decides the functions of the system for prototyping
27. Project Manager:

* Identifies the resources (hardware, software and Developer) for prototyping
* Prepares the plan and schedule
* Updates the Project Plan

1. The Project Manager or Designer briefs the system to the Developer
2. Project Manager or Designer verify the tools with the Developer and the Developer’s suggestion on different tool may be considered
3. Developer studies the User Interface Standards and performs the following activities:

* Paint the screens
* Paint the report formats
* Create tables / files, if a functional prototyping is required
* Code for screen navigation and the required functions
* Integrate the above components
* Test the integrated components

1. Project Manager and Designer verify the Prototype
2. Project Manager ensures the development of prototype falls through all SDLC process
3. The Prototype is demonstrated to the client or submitted for the client’s viewing
4. Project Manager ensures the client’s comments on the prototype is incorporated into the prototype and again demonstrated to the client or submitted for the client’s viewing. This happens in an iterative mode till the prototype is finalized
5. **Error Messaging Concept**
6. Describe the mechanism for displaying an error message to the user
7. Define the different error levels (information, warning, error, fatal error)
8. Describe how to display error messages and return codes from external (callable) systems
9. Where will the error texts be stored (plain ASCII text file, database, etc.)?
10. When will logs be written?
11. Document the design of the primary error-processing modules