HUIT Engineering Project Requirements Checklist

From HUIT Architecture Advisory Group

Contents

[hide]

Background and Purpose

Fundamental to making good investment and engineering decisions is clarity about the problem(s) to be solved by an engineering project. This is true whether it is a simple one person project or a project that will include many people over long periods of time. This article is intended to provide a set of requirements-related questions to be asked as people prepare for both the funding and engineering architecture, design, technology investigation, documentation, implementation and testing processes that would follow project approval.

How to use this Article

This is a checklist and not a static a fill in form. That is because not all projects are alike and this is not intended to substitute for engineering, product/project management and business judgement.

One suggested approach is to consider the nature of the project at hand and review this list and see which of the topic areas in this article are relevant to that project. From that review, you can create a shorter list to guide you through the requirements collection and management process.

Article Organization and Applicability

Note that this article is divided into sections in an effort to help focus on areas that apply to the project at hand. For example, if an infrastructure project to upgrade core elements of our network is being proposed, there may not be extended user interface requirements, but there may be many more details required about security and performance.

In short, this checklist and article that flow from it should be part of all engineering groups in HUIT, even if the code/technology does not actually face an end user.

Where does this Information Fit in the HUIT Process?

Whether a project is headed to the ITCRB/PRC for review or one that is not, the general principles outlined in this document apply. The project team members should review the material and tailor its application to the specifics of their program/project.

Requirements Sections

The following sections contain key areas that should be considered for each project. Not all may apply but is a section is omitted, understanding and being able to explain why would be helpful.

Project Justification - What is the Problem to be Solved?

Write a one or two paragraph summary that generally describes why the project is to be undertaken and the problems that are to be addressed as a result of undertaking the project. This should also tie HUIT and/or CIO Council goals and initiatives and specific benefits to HUIT and the University

Project Scope

As important as indicating what the project will do, it is important to communicate up front what is deemed to be outside the project scope. This can include:

1. Specific functions.
2. Integration with other systems that one might think are included.
3. Support for specific users, client types, etc.
4. etc.

The listing should include functions that have been specifically deferred to the next or subsequent release.

Team Organization and Functions

Identify the team members and functions, service owner, etc.. This is especially appropriate where more than one organization is involved. This section should clearly indicate each member and group's high-level responsibilities and to whom they are responsible in the project as well as for other things where there is a difference. That is, if a person is a member of a virtual team, the team leader may be thought to be the person the 'report to' for the project, but their actual manager is different. This information can help highlight those cases where there may be priority or other conflicts with the project.

Customer/User Requirements

This section is the bulk of the requirements document:

1. Functions in detail.
2. Performance expectations
    1. From a user experience perspective
    2. Aggregate requirements
3. Security requirements - user interface

4. Reporting requirements
    1. Standard reports
    2. Logging or security audits
5. Service Level Objects and agreements
    1. Notices of Failure
    2. Requirements in terms of time for service restoration
    3. MTBF (Meantime Between Failure) Requirements - availability etc.
6. Installability, Maintainability/Upgrade
7. Self maintenance requirements
    1. End users like PIN Self Maintenance or AutoReg.
    2. Users of our interfaces

Use Cases/User Scenarios

Use cases convey important information not apparent in even the most meticulously developed set of functional requirements. That said, they are not a replacement for a detailed set of functions that may not be presented in a use case.

Security Requirements

1. Threat identification and risk assessment
2. Special considerations for data at rest as well as in transit
3. Data accessibility
    1. Roles
    2. Read/write/modify and delete rules.
    3. Organizational rules
    4. Network access rules/restrictions
    5. Special considerations for HRCI or other sensitive data

Identity and Access Management

1. What type of authentication is required by the system?
2. Are authorization functions needed and for what 'roles'?

What HUIT systems have been evaluated and do they meet the requirements? If not, what is needed?

User Interface and Related Requirements

1. UI constraints
2. Mobile platform requirements
3. ADA Rules

Architectural Requirements

This section will detail how the proposed system will be constructed to meet the requirements at a high-level. It should reveal how availability/redundancy and other key aspects of the system will be met. Other considerations include:

1. Modularity requirements for integration with other systems.
2. Data management and archival
    1. Update requirements - units of granularity, time and ‘number of elements’
    2. Time to keep information on line
    3. Time to retain information.
    4. Requirements as the related to distribution and what data needs to be kept up to date and how often and how this relates to failover scenarios.
    5. Operating environment requirements and assumptions
3. Cloud issues, uptime requirements

Manageability (operational)

1. Faults (what to look for)
2. Configuration flexibility (what operational parameters can be changed).
3. Performance monitoring and alerting
4. Service level monitoring and altering
5. Accounting/logging requirements including ACL control and storage
6. Security monitoring for events/attackers

Capacity and Scalability

1. Number of concurrent users
2. Number of connections/unit time
3. Data throughput
4. Estimates of time of day, day of week, etc peak.
5. Public and Private software interface requirements
6. Disaster requirements e.g., when in DR Mode what are the requirements of the system.
7. High Availability and Redundancy Requirements

Quality of Service Related Requirements

This section outlines a number of dimensions related to service levels. It will be instructive in terms of the performance requirements previously list as well as in terms of what will be monitored. It should also enumerate issues related to:

1. Business criticality and requirements for continuous operation (or not).
2. Performance expectations in a degraded condition.
3. Trade-offs/considerations related short or long term outages, frequency and time sensitivity. For example we want the payroll system to run on time 12 months a year every week. Certain student systems are critical a specific times of the year.
4. Does the system have to support hardware and software upgrades while running without disrupting or degrading service?
5. Specific performance requirements in aggregate and from a user perspective.

Support/Education/Communication/Training Requirements

1. For end users - like students, faculty, etc.
2. For other software engineers that integrate our technology in their environment.
3. Requirements of the technology provider of the software we use in our projects.
4. Requirements/materials for the help desk and other support personnel.
5. What are the requirements for communication to other members of the technical community and customers?
6. Will special support be required for helping customers transition to the new technology?

Documentation

1. Engineering - internal.
2. Operations
3. Support
4. Customer
5. External Engineers.
6. Other on-line resources like FAQs

Audit Requirements

This should include not only security but also legal and university audit concerns.

Questions and Answers

This section contains questions and answers that will be updated as we learn more:

- When is the requirements document based on this checklist going to be required as part of the PRC CTO review process?

  It will take some time before people can be made aware of this checklist and can integrate it into their process. As a result, there is not likely to be a single cutover date, but specific projects may be asked to work on requirements, even if they are already through the PRC process and approved.

- When should I begin the requirements document based on this checklist?

  A key part of any effort should include the documentation of features/functions to be developed and assumptions that underly the project and the technical decisions made. Work in this area should begin at product inception. This approach will improve the likelihood that nothing slips through the cracks and makes completion of the report something that is integrated into day to day activities rather than an additional task added inserted at crunch time.

- Should a formal review of documents based on this checklist and other technical materials take place before the deadline for availability in advance of a PRC meeting?

  As noted in the previous question, it is better to start this process at project inception. Similarly, review of documents often and early can reveal problems while their repair will not impact schedule and cost too much. As a result, the team should have regular reviews of these documents throughout the entire process. Additionally, we are in a fluid environment and requirements, assumptions and technologies can change, even in the planning phases - this checklist and people's familiarity with it can be a good way to keep project members up to date and on the same page.

- I am used to seeing different requirements documents for business, project, functions, etc. Why is there just this one checklist?

  This checklist is not organized that way since we wanted to keep overhead as small as possible and not require multiple documents, some of which may not be appropriate for the HUIT environment.

- But we are using an Agile approach, do we really need this?

Yes! An Agile approach requires an understanding of the problem to be developed just as much as any other approach. Indeed since the ‘sprints’ are of a short duration and teams are more flexible, it is more important that everyone be clear about the problems to be solved by the effort. Just as with engineering a project layed out in longer phases, say months instead of weeks, at different times (sprints) there may be a greater or lesser emphasis on one type of activity than another. In the early parts of a project engineers may be setting up their development environments and working with project leaders developing use cases. Over time (and sprints) that work on use cases and basic setup may diminish. It is also the case that new requirements or technologies my cause, for a period, renewed effort on use cases, architectural review and other tasks.

- We don’t have the time?

  A truism in engineering activities has existed for some time and is generally well understood. That is, architectural defects are the most costly to repair followed by design then simple coding errors. It could be said that misunderstanding requirements is the most costly error we can make. It is a false economy not to do this work. This approach is not intended to force every project through go through this process. It is intended to raise an awareness of the issue and make everyone aware of the risks so that if a decision is made to ignore the process based on some other factors (and ignore good engineering practice) everyone will be aware of these factors and assume the risk.

- Collecting requirements into a static document seems like such an out of date approach?

  There is nothing static implied in this set of recommendations. Requirements and projects do evolve. Documenting them from the outset provides a reference point for change for the customers and the engineering effort. Periodically the requirements should be reviewed and updated.

- Where can I get help or provide feedback?

  HUIT Enterprise Architecture

Appendix - A References

What follows are some references that may be helpful to you as you develop your requirements. Some have been used in the preparation of this checklist:

InfoTech - a bit over detailed but worth a read through. http://www.infotech.com

Use Case Template http://www.infotech.com/research/use-case-template

Retrieved from "https://wikis.fas.harvard.edu/huitarch/HUIT_Engineering_Project_Requirements_Checklist"