

Non-Functional requirements template

**VERSION 1.0**

This template was created to enable departments to more easily develop their project plans. The Department of Technology, Consulting and Planning Division, created this template based on its experiences. The template relies on industry best practices combined with decades of experience on California state information technology projects. The way it was structured is to enable a department to complete the information related to its project without having to write background information related to the discipline. A department may use as much or as little of the template as it wishes.

**Template Instructions:**

* ***Instructions for completing*** this template – written for the author of the project plan - are encased in **[ ]** and the text is ***italicized*** *and* ***bolded.***
* *Examples* are provided as a guideline to the type of sample information presented in each section and the text is *italicized*.
* Boilerplatestandard language for each section is written in the document font and may be used or modified, as necessary.
* A department’s project specific information goes within the brackets ***<< >>***.
* *Informational text is italicized* within square brackets [ ] for informational purposes to the person who has to create the plan and includes background information, explanation, rationale, etc.

DOCUMENT HISTORY

| **DOCUMENT REVISION HISTORY** | | | |
| --- | --- | --- | --- |
| Date | Document Version | Revision Description | Author |
| 12/09/2014 | 1.0 | Initial Version | R Peterson |
|  |  |  |  |
|  |  |  |  |

TABLE OF CONTENTS

1. INTRODUCTION 1

2. OVERVIEW 1

3. REFERENCED DOCUMENTS 3

4. Non-functional requirements 3

4.1. Performance 4

4.1.1. User Responsiveness 4

4.1.1.1 Normal User Load 4

4.1.1.2 End-of-Month Load 5

4.1.2. Batch Processing 5

4.2. security 6

4.2.1. Auditing 6

4.2.1.1 HIPPA Data Access Auditing 6

4.2.1.2 Financial Auditing 7

4.2.2. Etc. 8

4.3. Compatibility 8

4.3.1. Cross-Browser Compatibility 8

4.3.2. Legacy Interface Compatibility 9

4.3.3. Etc. Compatibility 9

4.4. Maintainability 10

4.4.1. Error Reporting 10

4.4.1.1 Application Errors 10

4.4.1.2 Database Errors 10

4.4.1.3 Batch Errors 10

4.4.1.4 Etc. Errors 10

APPENDIX A. CONSIDERATIONS A-1

# INTRODUCTION

Non-functional requirements are defined and described in a variety of different ways, in a number of different sources (see references for a sample). In general, on-functional requirements identify the constraints upon the system and software by 1) identifying environmental conditions[[1]](#footnote-1) for the system, 2) the qualities the system must possess, and 3) all other constraints applicable for the development and implementation of the system. Non-functional requirements do not describe what the system and software will do in terms of the system or software’s functionality or behavior, nor do they identify deliverables or other transitional needs.

This document specifies the non-functional requirements for the system that have been elicited from the <<Enter Project Name>> Project stakeholders. These non-functional requirements were then analyzed, elaborated, and formatted into a standard requirements structure, and then re-reviewed and validated by the stakeholders to ensure the defined requirements are correct, consistent, and necessary. The details of the process used are defined and described in the Requirements Definition Plan.

# OVERVIEW

Each non-functional requirement specified in this document consists of five sections: Scenario, Requirement, Constraints, Verification Method, and Notes. For each non-functional requirement, all five sections must exist and each section must be consistent with the other sections as well as consistent across all non-functional requirements. The purpose and content of each section is described below.

The Scenario section “sets the stage” or defines the context for the requirement. It identifies what is occurring within the system and external to the system, such as by actors within a Use Case context. For example, performance non-functional requirement Scenarios always characterize the load that will be applied to a system when a requirement will be measured (external aspect) and should also identify if the system is in its normal state or potentially some degraded performance state, (an internal aspect, e.g., batch processing has begun, half of the virtual servers are down, etc.). This section is where the environmental conditions for the system are documented, including what the system is being exposed to, specific to each non-functional requirement.

The Requirement section is the actual non-functional requirement that will be measured when the scenario begins, (e.g., the system shall have a response time of 1 second). This section is where the qualities that the system must possess are defined.

The Constraints section identifies all constraints upon the Scenario, Requirement, or Verification Method. A Constraint may be considered a limitation, a boundary, or other factor that must be considered when creating a solution that meets the identified Requirement. An example commonly used for performance requirements is that network communication performance beyond the scope of the Project (e.g., the Internet) shall not be considered as part of the required system response time. This type of constraint may significantly change the understanding of the Requirement by changing the boundary for where the performance requirement actually applies; this also has an impact on how the requirement will be verified. This section documents all constraints applicable for the development and implementation of the system, where the identified constraints are specific and in the context of the specified non-functional requirement.

The Verification Method section must identify the approach to how the requirement will be verified; there is no value to the state in specifying a requirement if that requirement cannot be verified. There is also a cost to the state for every requirement specified regardless of whether or not it can be verified. The Verification Method does not need to be specific with respect to the tools, resources, or other lower level details. However, it must identify a sound and reasonable method or approach that can be used to verify the requirement and the identified method must be consistent with and actually verify the specified requirement. The Project’s Test Manager/Team can provide valuable input in assisting to defining the verification method. If a Verification Method cannot be identified that will actually verify the identified Requirement, this is a significant indicator that the requirement is a poor requirement and that a different or alternate approach to defining the stakeholders’ needs should be taken to achieve the required results. For example, some maintainability requirements for custom code are difficult to test or verify; an alternative approach is to add measureable and verifiable requirements for the development process to help ensure the resulting source code is maintainable. This type of approach changes an unverifiable product requirement to a verifiable process requirement to achieve the desired result. This Verification Method section is a quality step to validate that the other sections of the non-functional requirement meet the criteria for specifying a testable requirement.

The Notes section is available to document any additional information necessary to communicate the state’s needs for the specific requirement. Defining and specifying requirements is all about communicating the stakeholders’ needs and the better those needs can be communicated, the higher the probability that there will be a consistent understanding by all stakeholders, including a vendor, on what the requirement actually means and its intent. The Notes section allows for further elaboration for increased communications and further understanding.

In order to establish consistency and to aid in communications, the following table is used to document each non-functional requirement. The “ID#” field is a unique number to identify one requirement from all others by simply stating its ID#. The “Name of the Requirement” text should be edited and a brief descriptive name for the requirement entered.

| ID# | Unique | “ Name of the requirement “ |
| --- | --- | --- |
| Scenario: | | * Who or what initiates the scenario * The event that initiates the scenario * The system or environmental conditions (e.g., normal operations, shutting down) * Which part of system, or whole, is involved * How is the system being stressed |
| Requirement: | | What noticeable event happens as a result of the scenario |
| Constraints: | | Limitations, boundaries, other conditions that must be considered |
| Verification Method: | | Describe how the Requirement can be tested and verified |
| Notes: | | Additional information to help communicate the needs. |

# REFERENCED DOCUMENTS

The following documents were referenced and used in creating this deliverable; they also provide definitions and descriptions of non-functional requirements that are used in this document.

* ISO/IEC/IEEE 24765:2010(E) Systems and software engineering – Vocabulary
* IEEE “Guide to the Software Engineering Body of Knowledge® (SWEBOK®) Version 3.0”, January 17, 2014
* IIBA “A Guide to the Business Analysis Body of Knowledge® (BABOK® Guide) Version 2.0”, March 31, 2009
* SEI Technical Report CMU/SEI-95-TR-021, “Quality Attributes”, December 1995
* “Understanding Quality Attributes in Software Architecture, 3rd Edition”, Len Bass, Paul Clements, Rick Kazman, Sep 25, 2012

# Non-functional requirements

The Non-Functional Requirements for the << Project name >> are grouped below in the following categories: *[Identify the project specific non-functional requirement categories below; the following are examples:]*

* Performance
* Security
* Compatibility
* Maintainability
* Interface
* Operational
* Resource
* Etc.

## Performance

[All system performance requirements should be listed in this section; the following are examples. The first subsection, User Responsiveness is a sub-category of Performance, which may have multiple scenarios below this sub-category. If no sub-category is needed then don’t include it. However, all requirements should be listed under their top-level category, such as Performance.]

1. User Responsiveness
2. Batch Processing
3. Etc.

### User Responsiveness

#### Normal User Load

| ID# | *NF-PU-01* | Normal User Load |
| --- | --- | --- |
| Scenario: | | * During normal processing days, (i.e., not at end of month or some unusual peak processing period), the users will be entering data into the system at rate shown in the following figure. This figure shows the normal load represented by requests to the legacy system throughout a “normal” day. * The system is in normal operating mode (i.e., there is no unusual processing, late batch runs, or any other non-normal demands). |
| Requirement: | | The system response time for user requests, as identified in the figure, shall be less than 1 second for any individual request during the processing day for 90% of all requests and shall not exceed 3 seconds for any request. |
| Constraints: | | * Network delays beyond the control of the project shall not be included in the measurement of 1 second; the 1 second time shall apply from the time the request arrives at the front-end of the system boundary until it exits that boundary and returns to the end user. * The figure shows requests to the legacy system based on the transactions built into the system. If the new system expands or contracts the individual transaction functionality, the load shall expand or contract accordingly. (For example, if a single legacy transaction request supplies two items of information and the new system requires two transaction requests to perform the same function, then the load for these types of transaction requests shall be doubled.) |
| Verification Method: | | This requirement can be verified by:   * Reviewing and adjusting the figure load curve to account for the increase or decrease in transaction requests based on the design of the new system. * Use a loading tool, Load Runner or similar, to simulate the load by injecting the load at the system boundary and measuring the response times. * Evaluate the results to verify that 90% of all requests were less than or equal to 1 second and no request was greater than 3 seconds. |
| Notes: | | The following figure is the average count of transactions received by the legacy system by transaction type for the hour following the time shown, (e.g., for TX# 18 there is an average of 23 transactions per hour occurring between 8:00 AM and 9:00AM; the raw data is available).  TX# |

#### End-of-Month Load

A requirement scenario would be placed here, including all five sections, to specify the required User Responsiveness during a peak period, such as end-of-month for this example.

| ID# | *NF-PU-02* | end-of-month load |
| --- | --- | --- |
| Scenario: | |  |
| Requirement: | |  |
| Constraints: | |  |
| Verification Method: | |  |
| Notes: | |  |

### Batch Processing

A requirement scenario would be placed here, including all five sections, to specify the required Batch Processing performance requirement(s), as in this example.

| ID# | *NF-PB-01* | Batch processing Performance |
| --- | --- | --- |
| Scenario: | |  |
| Requirement: | |  |
| Constraints: | |  |
| Verification Method: | |  |
| Notes: | |  |

**[Continue to add subsections as needed.]**

## security

[*All system security requirements should be listed in this section, for example:]*

1. Auditing
2. Etc.

### Auditing

#### HIPPA Data Access Auditing

| ID# | *NF-SA-01* | HIPPA Data Access Auditing |
| --- | --- | --- |
| Scenario: | | A user, authorized or not, accesses (creates, reads, updates or deletes) database records containing HIPAA data, regardless of system state.  The system is the database.  How is the system being stressed? |
| Requirement: | | The system shall detect and log all users, authorized or not, who access (create, read, update or delete) HIPAA data. |
| Constraints: | | The users must be identified by a unique user identification that is traceable back to the individual with the following exceptions:  Non-user initiated processing, such as scheduled batch processing and database backups, shall be identified by a unique identification means that allows tracing the access back to the software code that made the change. For example:  Interface batch processing changes shall be identified by the batch process number traceable back to the batch program that accessed the data.  Database backup processes shall be identified by the backup process name.  The HIPAA access log and records shall not be able to be deleted though it shall be able to be archived in accordance with the database archiving requirement. |
| Verification Method: | | This requirements can be verified by:  Accessing HIPAA data through each of the application screens using each security role that has access to the screens and verify that a HIPAA access record was created for each access.  Accessing the data without using the application but using all other security roles that have access to the HIPAA data, such as DBA, System Administrator, Network Administrator, etc.  Identify all software programs that access HIPAA data by reviewing the source code for references to the database tables, execute each, and verify the HIPAA access record identifies each software program correctly, with sufficient information to trace the access to the program executed. |
| Notes: | | A separate functional requirement exists for a HIPAA Access Report, NF-RP-01, will also be used to verify this requirement in conjunction with accessing and reviewing the HIPAA access logs directly. |

#### Financial Auditing

| ID# | *NF-SA-02* | Financial Auditing |
| --- | --- | --- |
| Scenario: | |  |
| Requirement: | |  |
| Constraints: | |  |
| Verification Method: | |  |
| Notes: | |  |

### Etc.

| ID# | *NF-SE-01* | ETC. |
| --- | --- | --- |
| Scenario: | |  |
| Requirement: | |  |
| Constraints: | |  |
| Verification Method: | |  |
| Notes: | |  |

## Compatibility

***[All system Compatibility requirements should be listed in this section; the following are examples:]***

1. Cross-Browser Compatibility
2. Legacy Interface Compatibility
3. Etc. Compatibility

### Cross-Browser Compatibility

| ID# | *NF-CB-01* | cross-browser compatibility |
| --- | --- | --- |
| Scenario: | | * Public users will have the ability to access the public-facing components of the system and there is no control over the browsers that they may be using. When the public user accesses the system, the pages displayed shall be rendered without failing to display the pages consistently across all identified browsers. * The system is in normal operations mode (i.e., there are no problems identified with the public-facing web servers or other components necessary to render a page). |
| Requirement: | | * The public-facing components of the system shall be compatible with Microsoft Internet Explorer 8.0 and later and FireFox 4.0 and later. * If a public user is attempting to access the system without a compatible browser, the system shall display a message to the user that identifies the compatibility requirements necessary to access the system. |
| Constraints: | | The system shall not attempt or rely on the ability to download “add-in” that will be required to execute of the public user’s client machine. |
| Verification Method: | | This requirement can be verified by:   * Creating test stations with each type of required browser and executing a set of test scripts where the results shall be compared against each other to verify consistency and compatibility. * Or, use a Browser Compatibility Tool to verify compatibility and consistency. * To verify non-compatibility, use either: a) a Browser Compatibility Tool and review the exceptions, or b) install non-compatible browsers and verify the compatibility requirements are displayed **and** review the downloaded code to verify that the method used to identify non-compatible browsers is generic and will work for any browser. |
| Notes: | | Compatibility and consistency apply for all components that will be displayed on a page, to include size, color, positioning, effects, and all other attributes associated with the generation of Web page content. From a public user’s perspective, the rendered page should look identical regardless of the browser used. |

### Legacy Interface Compatibility

| ID# | *NF-CB-01* | Legacy Interface compatibility |
| --- | --- | --- |
| Scenario: | |  |
| Requirement: | |  |
| Constraints: | |  |
| Verification Method: | |  |
| Notes: | |  |

### Etc. Compatibility

| ID# | *NF-CB-01* | etc. compatibility |
| --- | --- | --- |
| Scenario: | |  |
| Requirement: | |  |
| Constraints: | |  |
| Verification Method: | |  |
| Notes: | |  |

## Maintainability

***[All system Maintainability requirements should be listed in this section, as in this example.]***

1. Error Reporting

### Error Reporting

#### Application Errors

#### Database Errors

#### Batch Errors

#### Etc. Errors

***[Repeat this approach for all of the applicable non-functional requirements for the project.]***

1. CONSIDERATIONS

| **CONSIDERATIONS FOR COTS, MOTS, and CUSTOM IMPLEMENTATION** | |
| --- | --- |
| **COTS** | * For COTS products, many of the non-functional requirements may not apply as they are already built into the COTS application and there is little that can be changed. However, external to the application, there are still a number of non-functional requirements that are applicable, which are those involving the environment where the COTS application is hosted. Therefore, non-functional requirements such as performance, availability, some security, and others are applicable and should be specified. |
| **MOTS** | * For MOTS, all parts of the application that is not modified, the non-functional requirements are limited, as in COTS. Also, specifying non-functional requirements related to items such as maintainability of code, security of the application, etc., the effect of these non-functional requirements is limited as they would only apply to the modified parts of the application. Therefore, MOTS should follow the approach of COTS when specifying non-functional requirements unless the project determines that there is value in adding non-functional requirements that only apply to part of the software application. |
| **CUSTOM** | * For custom developments, all non-functional requirements apply and should be specified consistent with the needs of the stakeholders and the project. |

1. “Environmental conditions” are not just limited to atmospheric type conditions but all conditions to which the system is exposed yet must still function. For example, public-facing (Internet) systems are exposed to attacks to gain access, deny services, etc.; this exposure is considered as an environmental condition due to the need for the system to withstand the attacks and survive when exposed. [↑](#footnote-ref-1)