NOAA NESDIS
CENTER for SATELLITE APPLICATIONS and RESEARCH

TRAINING DOCUMENT

TD-9
PROJECT REQUIREMENTS TRAINING DOCUMENT

Version 3.0

Hardcopy Uncontrolled

TITLE: TD-9: PROJECT REQUIREMENTS TRAINING DOCUMENT VERSION 3.0

AUTHORS:

Ken Jensen (Raytheon Information Solutions)

PROJECT REQUIREMENTS TRAINING DOCUMENT
VERSION HISTORY SUMMARY

Version	Description	Revised Sections	Date
1.0	No version 1		
2.0	New Training Document (TD-9.1) adapted from CMMI guidelines and Raytheon Systems Engineering practices by Ken Jensen (Raytheon Information Solutions)	New Document	11/30/2007
3.0	Renamed TD-9 and revised by Ken Jensen (RIS) for Version 3	All	10/1/2009

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF ACRONYMS

CDR	Critical Design Review
CICS	Cooperative Institute for Climate Studies
CIMSS	Cooperative Institute for Meteorological Satellite Studies
CIOSS	Cooperative Institute for Oceanographic Satellite Studies
CIRA	Cooperative Institute for Research in the Atmosphere
CL	Check List
CM	Configuration Management
CMMI	Capability Maturity Model Integration
COTS	Commercial Off The Shelf
CPU	Central Processing Unit
CREST	Cooperative Remote Sensing and Technology Center
DG	Document Guideline
DPP	Development Project Plan
EPL	Enterprise Product Lifecycle
IPT	Integrated Product Team
MOU	Memorandum of Understanding
NESDIS	National Environmental Satellite, Data, and Information Service
NOAA	National Oceanic and Atmospheric Administration
OCD	Operations Concept Document
PAR	Process Asset Repository
PDR	Preliminary Design Review
PG	Process Guideline
PP	Project Proposal
PRG	Peer Review Guideline
PRR	Project Requirements Review
QA	Quality Assurance
QFD	Quality Function Deployment
RAD	Requirements Allocation Document

RAS	Requirements Allocation Sheet
RHTM	Requirements Horizontal Traceability Matrix
RNM	Requirements/Needs Matrix
RVTM	Requirements Vertical Traceability Matrix
SEI	Software Engineering Institute
SG	Stakeholder Guideline
SOW	Statement of Work
SRR	System Readiness Review
STAR	Center for Satellite Applications and Research
STP	System Test Plan
SWA	Software Architecture Document
TD	Training Document
TG	Task Guideline
TRR	Test Readiness Review
UTP	Unit Test Plan
VVP	Verification and Validation Plan
VVR	Verification and Validation Report

1. INTRODUCTION

The NOAA/NESDIS Center for Satellite Applications and Research (STAR) develops a diverse spectrum of complex, often interrelated, environmental algorithms and software systems. These systems are developed through extensive research programs, and transitioned from research to operations when a sufficient level of maturity and end-user acceptance is achieved. Progress is often iterative, with subsequent deliveries providing additional robustness and functionality. Development and deployment is distributed, involving STAR, the Cooperative Institutes (CICS, CIMSS, CIOSS, CIRA, CREST) distributed throughout the US, multiple support contractors, and NESDIS Operations.

NESDIS/STAR is implementing an increased level of process maturity to support the exchange of these software systems from one location or platform to another. The Project Requirements Training Document (TD) is one component of this process.

1.1. Objective

This TD is a STAR EPL process asset that includes techniques and tools for the development of project requirements, their allocation to system components and product components, their maintenance, and their management throughout the product lifecycle. It includes guidelines for the identification of project requirements from customer/user needs and expectations, guidelines for requirements analysis, guidelines for the quality assurance (QA) of project requirements (including traceability, tracking, validation and configuration management (CM)), guidelines for allocating requirements to system components and product components as the design is developed, guidelines for updating the requirements and requirements allocation, and guidelines for requirements management.

The objective of this TD is to provide training for the project requirements process. This process includes requirements identification, analysis, allocation, tracking, validation, verification, maintenance, and management. The intended users of this TD are the stakeholders in the project planning and requirements development steps of the STAR Enterprise Product Lifecycle (EPL)[1], including managers, developers, reviewers and QA personnel. Customers and end users of products developed through the STAR EPL will also benefit from a familiarity with the concepts described in this TD, particularly if they are involved with the product lifecycle.

[1] A description of the STAR EPL can be found in STAR EPL Process Guidelines (PG-1, c.f. Section 2 of this TD).

1.2. Background

This TD has been adapted from Capability Maturity Model Integration (CMMI) practices (CMMI-DEV-v1.2, 2006) and Raytheon systems engineering practices. It has been tailored to fit the STAR EPL process.

1.3. Overview

This TD contains the following sections:

Section 1.0 -	Introduction
Section 2.0 -	References
Section 3.0 -	The Project Requirements Step
Section 4.0	Requirements Identification
Section 5.0 -	Requirements Analysis
Section 6.0 -	Requirements Allocation
Section 7.0 -	Requirements Quality Assurance
Section 8.0 -	Requirements Maintenance
Section 9.0 -	Requirements Management
Appendix A -	Examples

NOTE: In the sections to follow, you will notice items that are intended to be “DO” directions for project stakeholders who are involved in one or more requirements development tasks. The intention of these items is to clearly highlight items that project stakeholders are expected to do. These items are partitioned, in bold font, and marked with a special bullet, like this example:

Ordinary content

- **DO this.**

More ordinary content.

The remainder of the content of this TD, the “ordinary” content, is information to support the stakeholders in accomplishing the “DO” directions.

The reader of this TD is urged to read the “DO” items in each subsection and then go back and read the entire subsection with an understanding of what should be done to accomplish the purpose of the functions described in that subsection.

2. REFERENCE DOCUMENTS

CMMI-DEV-v1.2 (2006) contains the goals and practices of the current Capability Maturity Model Integration (CMMI) of the Carnegie-Mellon Software Engineering Institute (SEI). This document can be accessed from the SEI web site (http://www.sei.cmu.edu/publications/documents/06.reports/06tr008.html).

All of the following references are STAR EPL process assets that are accessible in a STAR EPL Process Asset Repository (PAR) on the STAR web site:

http://www.star.nesdis.noaa.gov/star/EPL_index.php.

PG-1: STAR EPL Process Guideline provides the definitive description of the standard set of processes of the STAR EPL.

PG-1.A: STAR EPL Process Guideline Appendix, an appendix to PG-1, is a Microsoft Excel file that contains the STAR EPL process matrix (Stakeholder/Process Step matrix), listings of the process assets and standard artifacts, descriptions of process gates and reviews, and descriptions of stakeholder roles and functions.

TG-6: Project Requirements Task Guideline provides a description of standard tasks for process step 6, during which the project requirements are developed.

TG-7: Preliminary Design Task Guideline provides a description of standard tasks for process step 7, during which the project requirements and requirements allocations are developed and refined.

TG-8: STAR EPL Detailed Design Task Guidelines provides a description of standard tasks for process step 8, during which the project requirements and requirements allocations are developed and refined.

DG-6.1: Operations Concept Document Guidelines contains the guidelines for writing a project Operations Concept Document (OCD).

DG-6.2: Requirements Allocation Document Guidelines contains the guidelines for writing a project Requirements Allocation Document (RAD).

DG-6.3: Verification and Validation Plan Guidelines contains the guidelines for writing a project Verification and Validation Plan (VVP).

DG-11.4: Verification and Validation Report Guidelines contains the guidelines for writing a project Verification and Validation Report (VVR).

TD-6: Verification and Validation Training Document provides techniques and tools for verification and validation.

3. THE REQUIREMENTS DEVELOPMENT PROCESS

Project requirements are the skeleton that holds together all aspects of a project. Any stakeholder working on any project task should be able to answer this question – why am I doing this? The answer should be – to satisfy requirement X, Y, and/or Z. A stakeholder should be able to track any task back to its driving requirements, which should be documented in the project RAD (c.f. Section 4.7 of this TD). Without approved and documented requirements, there is no assurance that any stakeholder understands the project.

The requirements development process is structured to take very broad statements of mission needs and develop refined system-specific operational requirements from which performance specifications can be derived.

Project requirements are developed, refined and documented during a defined phase of the STAR EPL, the Design Phase, as illustrated in Figure 3.1.

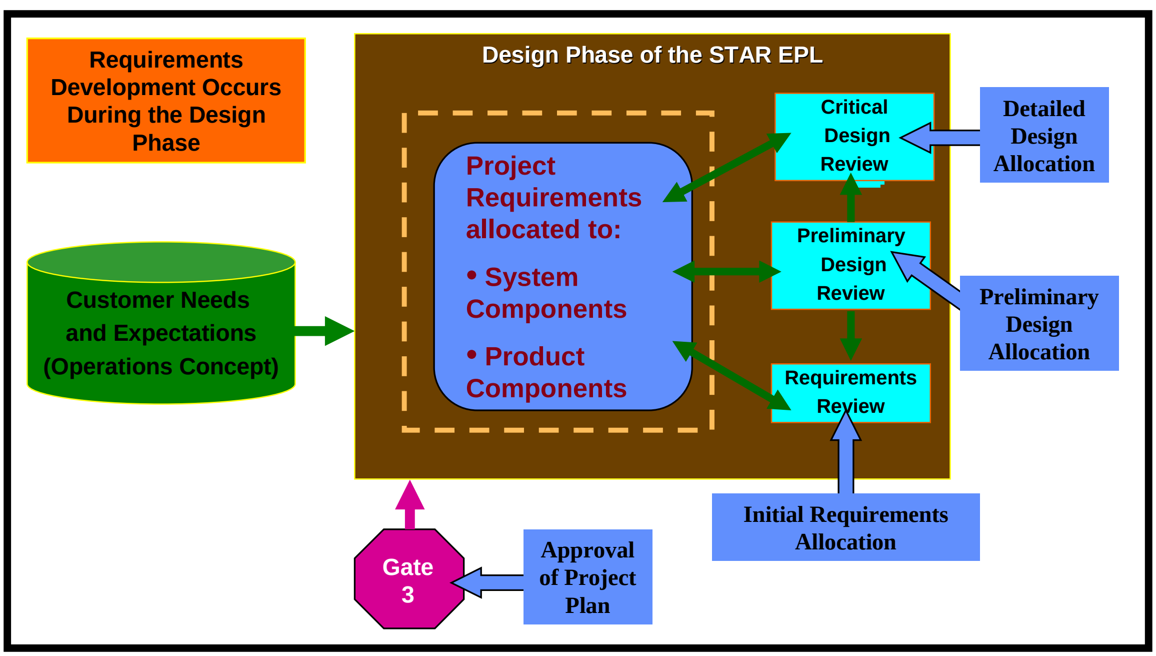

Figure 3.1 – Requirements Development Process

The Design phase commences when the project passes the Gate 3 Review. At this point, a Development Lead and an Integrated Product Team (IPT) have been identified and a resource-loaded schedule of project tasks has been created. Included among these tasks will be the tasks associated with requirements development that are described in this TD.

As Figure 3.1 illustrates, the project requirements are allocated to system components and product components of the product processing system. Requirements allocation will be discussed in Section 6 of this TD.

This phase includes three steps:

- Project Requirements (step 6 of the STAR EPL)
- Preliminary Design (step 7 of the STAR EPL)
- Detailed Design (step 8 of the STAR EPL)

The primary purpose of step 6 is to develop a set of project requirements in sufficient scope and detail to allow for the design of a product processing system that will deliver products that satisfy the end users. Step 6 culminates with a Project Requirements Review (PRR)[2].

The primary purpose of step 7 is to develop a preliminary design for each alternative solution that is being considered and to allocate the requirements to each product and system component in the preliminary design. Step 7 culminates with a Preliminary Design Review (PDR).

The primary purpose of step 8 is to develop the detailed design for a preferred solution and to allocate the requirements to each product and system component in the detailed design. Step 8 culminates with a Critical Design Review (CDR), followed by a Gate 4 Review.

As the project matures throughout the Design phase, an increasingly comprehensive and mature requirements allocation is reviewed at each of the three technical reviews (PRR, PRD and CDR) of this phase.

[2] Refer to the STAR EPL Process Guidelines (PG-1 and PG-1.A) for a description of the STAR EPL gates and reviews.

The primary requirements development tasks to be accomplished during steps 6-8 are:

- Requirements Identification (c.f. Section 4 of this TD).
- Requirements Analysis (c.f. Section 5 of this TD).
- Requirements Allocation (c.f. Section 6 of this TD).
- Requirements Quality Assurance (c.f. Section 7 of this TD).

These tasks are nominally performed in sequence, with Identification preceding Analysis, Analysis preceding Allocation, and Allocation preceding Quality Assurance. In practice, there will usually be a significant amount of feedback between these tasks, resulting in an iterative requirements development process. Figure 3.2 shows a high level Requirements Development process flow that illustrates these characteristics.

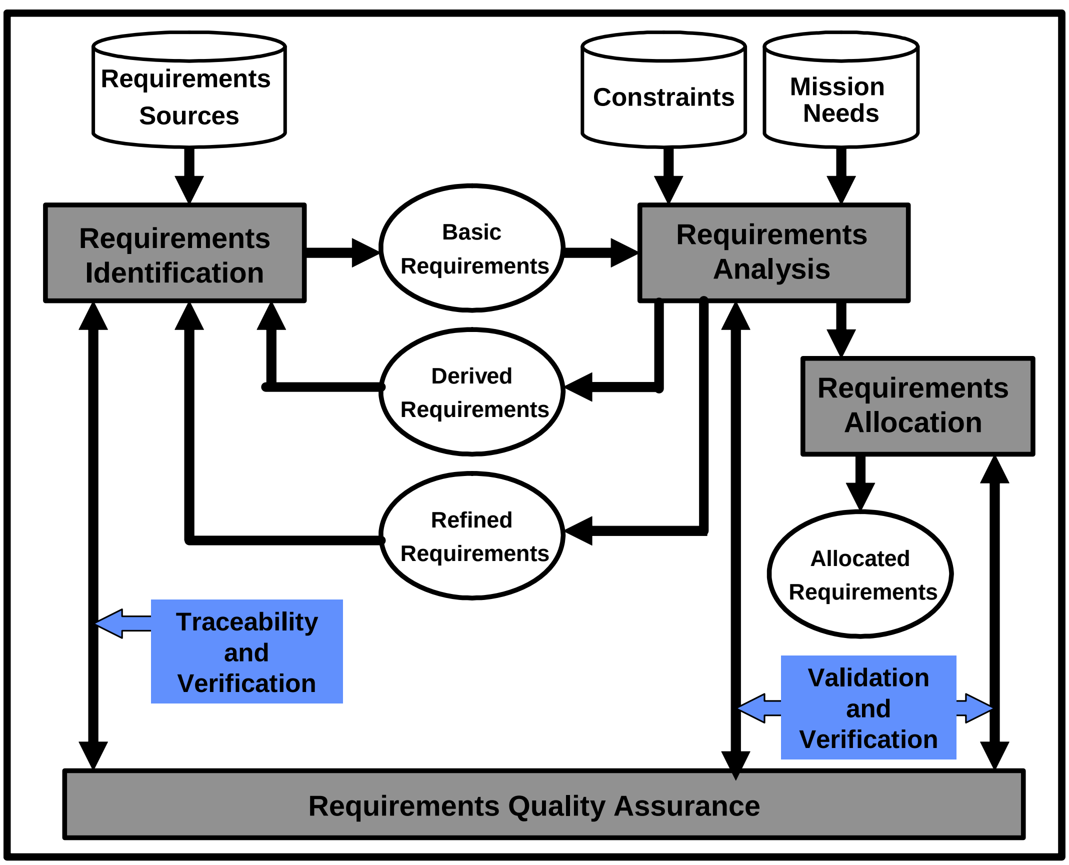

Figure 3.2 – Requirements Development Process Flow

As shown in Figure 3.2, inputs to the process are Requirements Sources, Constraints, and Mission Needs. Outputs are the requirements and their allocation.

Figure 3.2 also shows the feedback loops and iteration that are characteristic of the process. This works as follows:

- Basic requirements are identified from the customer/user needs and expectations obtained from the approved requirements sources (c.f. Section 4.1 if this TD).
- Basic requirements are analyzed to determine derived requirements and to refine basic and derived requirements. The analysis (c.f. Section 5 of this TD) uses system constraints and the NESDIS strategic and mission plan as inputs to produce the derived and refined requirements. This iterative loop between identification and analysis continues throughout the Design phase to produce an increasingly comprehensive and mature set of requirements.
- Requirements are allocated to system and product components as the design matures.
- Quality Assurance of the requirements and their allocation is performed throughout the process. Requirements QA includes Traceability (c.f. Section 7.1 of this TD) of identified requirements, Validation (c.f. Section 7.3 of this TD) of the requirements analysis and allocation, and Verification (c.f. Section 7.4 of this TD) of all requirements development sub-processes.

These sub-processes will be discussed in detail in the sections to follow.

It is important to realize that the set of project requirements will typically be refined after PRR, primarily in steps 7 and 8 as solutions and design mature. The essential iterative nature of the requirements development process, and its close connection to solutions and design development, is illustrated in Figure 3.3.

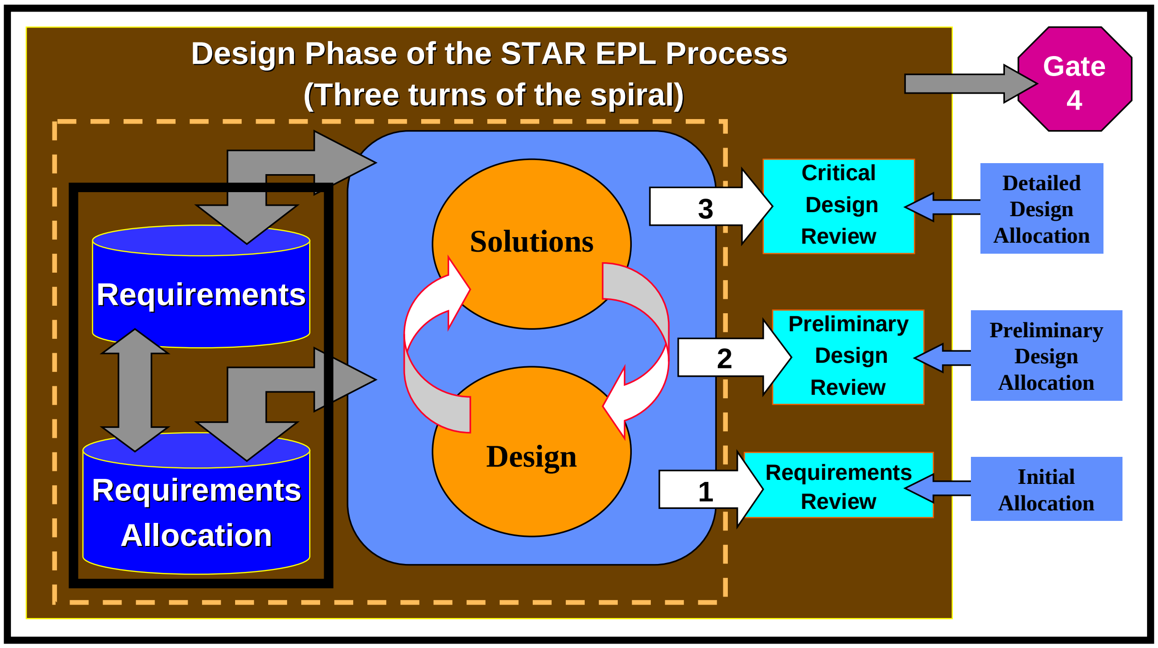

Figure 3.3 – Iterative Development of Project Requirements

Figure 3-3 shows the iterative interaction between the project requirements, their allocation to product components and system components, and the solutions and design that are developed to satisfy the requirements. Note that the process flow shown in Figure 3.2 fits into Figure 3.3 as the black-bordered region. These concepts will be described more fully in the sections to follow.

By the culmination of the Design phase, project requirements identification, analysis and allocation should be largely complete. Refinements after Gate 4 Review may be necessary due to unforeseen risks and actions that arise after this phase, but they should be minor. It is important that project management controls the process of refining requirements to avoid requirements creep (c.f. Section 9 of this TD).

The project developers should be in consultation with all customers and users while developing requirements. The various inputs from the customers must be consolidated, missing information must be obtained, and conflicts must be resolved in documenting the recognized set of customer requirements. Relevant stakeholders representing all phases of the product's life cycle should include business as well as technical functions. In this way, concepts for all product-related life-cycle processes are considered concurrently with the concepts for the products. Customer requirements result from informed decisions on the business as well as technical effects of their requirements. Verification and validation requirements are best refined by collaboration between the customer and the project developers (c.f. Section 7 of this TD).

4. REQUIREMENTS IDENTIFICATION

Requirements identification is a process of turning the customer/user needs and expectations into a specific set of requirements on the product processing system.

We begin with the STAR definition of what a requirement is. This is adopted from the CMMI glossary:

"(1) A condition or capability needed by a user to solve a problem or achieve an objective.

(2) A condition or capability that must be met or possessed by a product or product component to satisfy a contract, standard, specification, or other formally imposed documents.

(3) A documented representation of a condition or capability as in (1) or (2)."

Requirements are identified as process requirements, product requirements, or system requirements. The distinction between product and system requirements is useful for requirements management. This will be discussed in Section 9 of this TD.

The requirements identification process should consist of the following six sub-processes, described in the subsections to follow:

1) Identify the sources of requirements
2) Determine the requirements for the processes that will be used during the product lifecycle.
3) Develop an operations concept, based on customer/user needs and expectations
4) Develop basic product requirements
5) Develop basic system requirements
6) Develop derived requirements

The first two of these sub-processes, described in Sections 4.1 and 4.2 of this TD, should be accomplished during the Planning for Development Phase of the product lifecycle prior to the Project Requirements step.

4.1. Requirements Sources

The requirements identification process actually begins with the Plan phase of the STAR EPL, where the sources for requirements are identified.

To avoid requirements creep, criteria are established to designate appropriate channels, or approved sources, from which to receive requirements. PUSH users and PULL users will often be the source for basic requirements and the development IPT will typically be the source for derived requirements. *Potential* end users should be considered for approval during project planning and *approved* end users should be identified and included in the Development Project Plan (DPP).

All approved sources for requirements should be identified in the DPP. Requirements from other sources can be considered, but it should be noted when a requirement comes from an unapproved source and the customer must be fully informed when alternative sources of requirements are being considered.

Regardless of the source, all requirements must be approved at PRR and subsequent standard reviews before they are adopted.

- **Assemble a list of approved sources for project requirements. Refer to the DPP, which should document the approved sources. Confirm that you have the correct contact information for these sources.**

4.2. Process Requirements

The process to be followed by the project must be defined as a set of process requirements. Process requirements for a specific project may be tailored from STAR's set of standard processes (PG-1) using tailoring guidelines (PG-2). It is sufficient to note only what is tailored (i.e., it is not necessary or recommended to repeat the entire set of standard processes). If there are no tailored practices, note that the organization's set of standard processes is adopted. There should be one and only one basic process requirement. It should be numbered 0.0 and should be stated in one of the following two ways:

1) Requirement 0.0: "The STAR organization's set of standard processes shall be followed."
2) Requirement 0.0: "The STAR organization's set of standard processes shall be followed, except as specified in requirements 0.1, 0.2, etc."

The latter statement should be used if there is any tailoring of the standard processes to fit a project's unique needs. In that case, there should be derived requirements (as many and down to whatever level necessary) that describe the tailoring.

- **Identify basic process requirement 0.0. Refer to the DPP, which should state whether the STAR set of standard processes shall be followed and, if not, describe how the standard processes have been tailored to fit the project's unique needs. If there has been tailoring, identify specific tailored processes as derived process requirements (0.1, 0.2, etc.).**

-

Process requirements include:

1) Lifecycle steps. Note any tailoring from the standard STAR EPL steps, stated as a derived process requirement.

2) Gates and reviews. Note any tailoring from the standard STAR set of gates and reviews, stated as a derived process requirement.

3) Entry criteria for defined reviews. STAR standard entry criteria for each standard review are documented in the Peer Review Guideline (PRG) and Check List (CL) for that review. A project may tailor its entry criteria. Any tailored entry criteria should be obtained from the DPP. This should be stated as a derived process requirement, as follows:

   Requirement 0.x – "Entry criteria for gates and reviews shall be tailored from STAR's standard set of entry criteria."

Requirement 0.x.y – “The entry criteria for <Review name> shall be the tailored entry criteria that are stated in the DPP.”

OR

Requirement 0.x.y – “The entry criteria for <Review name> shall be <Explicit statement of the entry criteria>.”

Which way to state Requirement 0.x.y is at the discretion of the project. It is easier to refer to the DPP, but it may be better for requirements analysis and management to have the tailored entry criteria explicitly stated in the requirements documentation. For example, the desire for tailoring of the entry criteria for a specific review may arise after project planning. In that case, an explicit statement of tailored entry criteria in the requirements documentation will help to ensure that this change finds it way into the project plan.

4) Exit criteria for defined reviews. STAR standard exit criteria for each standard review are documented in the PRG and CL for that review. A project may tailor its exit criteria. Any tailored exit criteria should be obtained from the DPP. This should be stated as a derived process requirement, as follows:

Requirement 0.x – “Exit criteria for gates and reviews shall be tailored from STAR’s standard set of exit criteria.”

Requirement 0.x.y – “The exit criteria for <Review name> shall be the tailored exit criteria that are stated in the DPP.”

OR

Requirement 0.x.y – “The exit criteria for <Review name> shall be <Explicit statement of the exit criteria>.”

Which way to state Requirement 0.x.y is at the discretion of the project. It is easier to refer to the DPP, but it may be better for requirements analysis and management to have the tailored exit criteria explicitly stated in the requirements documentation. For example, the desire for tailoring of the exit criteria for a specific review may arise after project planning. In that case, an explicit statement of tailored exit criteria in the requirements documentation will help to ensure that this change finds it way into the project plan.

5) Stakeholder roles and responsibilities. STAR’s standard processes include general descriptions of stakeholder roles and responsibilities, but these often must be tailored for individual projects. It is essential that the DPP states any tailored stakeholder roles and responsibilities. Tailored stakeholder roles and responsibilities should be stated as a derived process requirement, as follows:

   Requirement 0.x – “Stakeholder roles and responsibilities shall be tailored from STAR’s standard set of processes.”

      Requirement 0.x.y – “Stakeholder roles and responsibilities shall be as stated in the DPP.”

   It is usually too cumbersome to capture the specific roles and responsibilities of stakeholders in the requirements documentation. It is therefore essential that the DPP accomplish this.

-

- **Ensure that the DPP states the roles and responsibilities of project stakeholders. Determine whether these have been tailored from the STAR standard stakeholder roles and responsibilities. If so, include a derived process requirement that states that stakeholder roles and responsibilities shall be as stated in the DPP.**

-

6) Task Guidelines (TG). Note any tailoring from the standard STAR TGs. If a specific TG is to be tailored, the project should produce a revised TG as a project artifact. In that case, the DPP should make an explicit reference to the revised TG and there should be a derived process requirement that states:

   Requirement 0.x – “Task guidelines shall be tailored from STAR’s standard set.”

      Requirement 0.x.y – “STAR standard TG-m.n shall be replaced by <Artifact Name>.”

-

➢ **Determine whether any task guidelines have been tailored from the standard STAR TGs. Refer to the DPP, which should state this and should identify a project artifact that contains the tailored task guidelines. If so, include a derived process requirement that states that the designated STAR standard TG shall be replaced by the designated project artifact.**

-

7) Stakeholder Guidelines (SG). Note any tailoring from the standard STAR Stakeholder Guidelines (SGs). If a specific SG is to be tailored, the project should produce a revised SG as a project artifact. In that case, the DPP should make an explicit reference to the revised SG and there should be a derived process requirement that states:

Requirement 0.x – “Stakeholder guidelines shall be tailored from STAR’s standard set.”

Requirement 0.x.y – “STAR standard SG-m.n shall be replaced by <Artifact Name>.”

-

➢ **Determine whether any stakeholder guidelines have been tailored from the standard STAR SGs. Refer to the DPP, which should state this and should identify a project artifact that contains the tailored stakeholder guidelines. If so, include a derived process requirement that states that the designated STAR standard SG shall be replaced by the designated project artifact.**

-

8) Document Guidelines (DG). Note any tailoring from the standard STAR DGs. If a specific DG is to be tailored, the project should produce a revised DG as a project artifact. In that case, the DPP should make an explicit reference to the revised DG and there should be a derived process requirement that states:

Requirement 0.x – “Document guidelines shall be tailored from STAR’s standard set.”

Requirement 0.x.y – “STAR standard DG-m.n shall be replaced by <Artifact Name>.”

-

➢ **Determine whether any document guidelines have been tailored from the standard STAR DGs. Refer to the DPP, which should state this and should identify a project artifact that contains the tailored document guidelines. If so, include a derived process requirement that states that the designated STAR STAR standard DG shall be replaced by the designated project artifact.**

-

9) Peer Review Guidelines (PRG). Note any tailoring from the standard STAR PRGs. If a specific PRG is to be tailored, the project should produce a revised PRG as a project artifact. In that case, the DPP should make an explicit reference to the revised PRG and there should be a derived process requirement that states:

Requirement 0.x – “Peer review guidelines shall be tailored from STAR’s standard set.”

Requirement 0.x.y – “STAR standard PRG-m.n shall be replaced by <Artifact Name>.”

-

➢ **Determine whether any peer review guidelines have been tailored from the standard STAR PRGs. Refer to the DPP, which should state this and should identify a project artifact that contains the tailored peer review guidelines. If so, include a derived process requirement that states that the designated STAR standard PRG shall be replaced by the designated project artifact.**

-

10) Review Check Lists (CL). Note any tailoring from the standard STAR CLs. If a specific CL is to be tailored, the project should produce a revised CL as a project artifact. In that case, the DPP should make an explicit reference to the revised CL and there should be a derived process requirement that states:

Requirement 0.x – "Review Check Lists shall be tailored from STAR's standard set."

Requirement 0.x.y – "STAR standard CL-m.n shall be replaced by <Artifact Name>."

-

- **Determine whether any review Check Lists have been tailored from the standard STAR CLs. Refer to the DPP, which should state this and should identify a project artifact that contains the tailored review Check Lists. If so, include a derived process requirement that states that the designated STAR standard CL shall be replaced by the designated project artifact.**

-

11) Project artifacts. Note any tailoring from the STAR standard list of project artifacts. If there has been tailoring, the DPP should contain a list of added and/or deleted artifacts and there should be a derived process requirement that states:

Requirement 0.x – "The list of required project artifacts shall be tailored from STAR's standard set."

Requirement 0.x.y – "<Artifact Name> shall be added/deleted."

-

- **Determine whether any project artifacts have been tailored from the standard STAR set of artifacts. Refer to the DPP, which should state this and should contain a list of added and/or deleted artifacts. If so, include a derived process requirement that states that the designated artifact shall be added or deleted.**

-

4.3. Operations Concept

A requirements generating system should be structured to take very broad statements of NESDIS mission needs, customer needs and end user needs, and develop refined system-specific requirements. To accomplish this, the development team should construct an operational scenario, based on customer/end user needs and expectations. This operational scenario, known as an operations concept, serves as a bridge between the requirements sources and the requirements.

An operational scenario is a description of an imagined sequence of events that includes the interaction of the product with its environment and users, as well as interaction among its product components. Operational scenarios are used to analyze the requirements and design of the system (c.f. Section 5.2 of this TD) and to verify and validate the system (c.f. Sections 7.3 and 7.4 of this TD).

- **Determine whether customers and approved end users have documented their needs and expectations (e.g., in a concept of operations document or in a Project Proposal (PP). If not, consult with them to obtain a clear understanding of their needs and expectations.**

The requirements developers should construct and document an operations concept in an Operations Concept Document (OCD). Refer to STAR EPL process asset DG-6.1 for OCD guidelines.

- **Develop an operations concept. Document in an Operations Concept Document (OCD) for review at the PRR, following guidelines in DG-6.1.**

4.4. Basic Product Requirements

A product is defined as a tangible item that is delivered to a customer or end user.

Product requirements include requirements on product content, performance, operational production (e.g. timeliness), and end use.

Product requirements include requirements on product components. Product components are items that are designed to be integrated to build a product. Requirements on product components are typically derived requirements (c.f. Section 4.6 of this TD), except for cases where:

1) The customer has explicitly identified a product component as a required part of the delivered product, or
2) The product component by itself satisfies an identified customer requirement.

The following are examples of types of requirements that are product requirements:

- “The product shall include an atmospheric vertical temperature profile”
- “The atmospheric vertical temperature profile shall contain at least 32 layers”
- “The reported temperature for each vertical layer shall have an accuracy of 0.5 K”
- “The atmospheric vertical temperature profile shall be produced operationally within 30 minutes of its observation”
- “The atmospheric vertical temperature profile shall be reported in a format that can be used as input to a NWP model at the NWS”

Basic product requirements address the satisfaction of customer needs, customer expectations, and project objectives derived from these needs and expectations or from the NESDIS mission and strategic plan

Figure 4.1 shows a context-level schematic of the sources of basic product requirements.

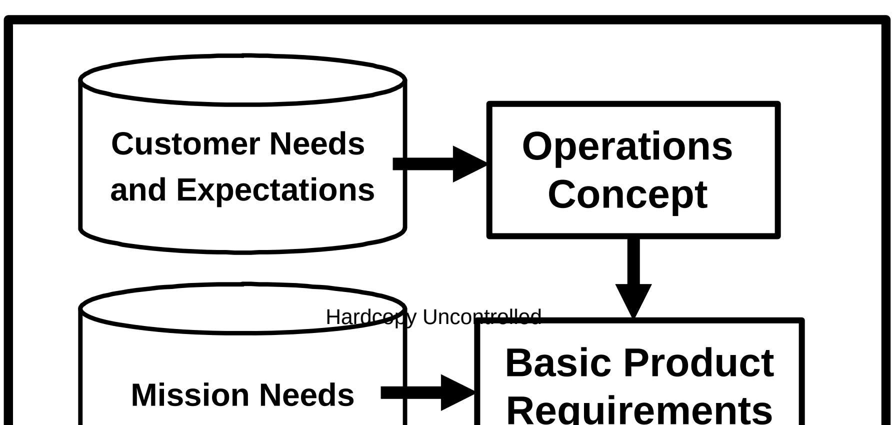

Figure 4.1 – Basic Product Requirements

Each basic product requirement should be traceable to the operations concept or the NESDIS mission and strategic plan. This trace should be identified as the driver for the requirement. The driver for each basic product requirement should be documented in the RAD (c.f. Section 4.7 of this TD).

-

- **Identify basic product requirements. Use the NESDIS strategic and mission plan and the operations concept as references. Document in RAD v1r0 for review at the PRR, following guidelines in DG-6.2.**

-

Product performance objectives should be captured in basic product requirements. The DPP may identify performance objectives, but these will be preliminary. Trades, risk assessment and solutions development are needed to refine the basic requirements that are driven by performance objectives. These requirements should be carefully tracked and reviewed at PRR, PDR and CDR, with customer/user visibility and buy-in.

4.5. Basic System Requirements

System requirements include system component characteristics (e.g. security, portability), interfaces and dependencies (e.g. code, test data, production environments and platforms).

Examples of system requirements include:

- External interface requirements
- Security requirements
- Development environment requirements
- Transition environment requirements
- Operational environment requirements
- Test environment requirements
- Monitoring and Maintenance requirements
- Delivery requirements

Basic system requirements are directly traceable to system constraints (e.g. security, portability, external interfaces) or basic product requirements. Derived system requirements are those that are not directly traceable to system constraints or basic product requirements. Derived system requirements should be traceable to basic system requirements (e.g. production environments and platforms).

Figure 4.2 shows a context-level schematic of the sources of basic system requirements.

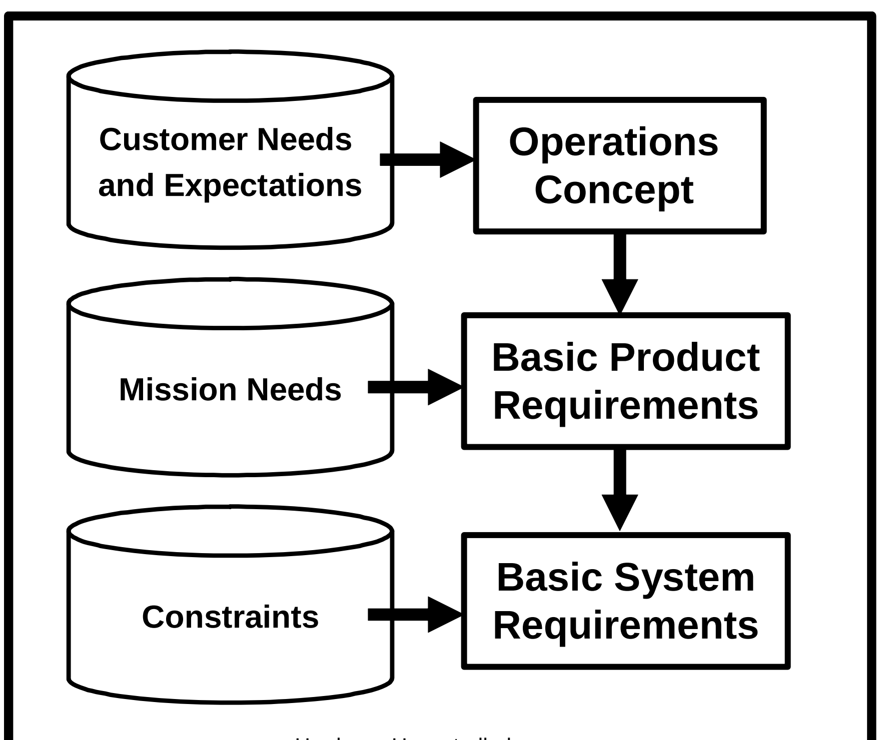

Figure 4.2 – Basic System Requirements

Each basic system requirement should be traceable to a basic product requirement or an identified constraint. This trace should be identified as the driver for the requirement, and documented in the RAD (c.f. Section 4.7 of this TD).

-

- **Identify basic system requirements, including requirements on external interfaces, security, software, development, transition, operational and test environments, delivery and maintenance. Use the basic product requirements and constraints identified during project planning as references. Document in the RAD for review at the PRR, PDR, and CDR, following guidelines in DG-6.2.**

-

External interface requirements will be refined during the Design phase, as the design matures. By CDR, a Software Architecture Document (SWA) will document all the external software interfaces. At CDR, the documentation of system requirements in the RAD should include all interfaces that are documented in the SWA. If the system design includes any system components that will be acquired from an external supplier, requirements pertaining to a supplier agreement must be developed. Often, these requirements will be documented in a Statement of Work (SOW), contract, or Memorandum of Understanding (MOU). If such a formal document exists, it is sufficient to include a requirement in the RAD that states that any and all requirements pertaining to <Supplier> shall be as documented in the <Supplier Agreement Document>. Under these circumstances, the formal supplier agreement must be maintained in the project baseline, either as a standalone document or as part of the DPP.

Security requirements should be developed in close consultation with the end users and the security office of the development and operational organizations. Security requirements can be complex and variable, depending upon the requirements they are derived from. For example, security requirements for an operational environment will generally be stricter than security requirements for a development environment.

Software requirements are distinct from software functionality and software design. Software requirements address whether software will be acquired, re-used or developed. They also address restrictions (e.g. COTS) and coding standards. For example, a software requirement may state that all product processing code must be C++. Note that the STAR EPL standard processes contain organizational coding standards that are documented in the process assets. If a project wishes to tailor STAR EPL coding standards, it must be stated as an explicit requirement. The PRR will require that software requirements are clearly stated in the RAD. The Test Readiness Review (TRR) of the Build phase will require that pre-operational code meets the project's software requirements.

Although software requirements are commonly considered to be system requirements, there are cases where they will be product requirements. These cases occur when the software item in question will be delivered to end users, either as a standalone item or integrated into a delivered product. A common example is product processing code that will be delivered to an operations center.

Development environment requirements are requirements on the system used for the Build phase of the STAR EPL. This phase includes pre-operational code development, pre-operational test data development, and system integration. Requirements should include specific hardware requirements (e.g. CPUs, platforms, location). Often, the development environment will be determined during project planning. In that case, the DPP should document this, and the requirements developers should ensure that the development environment plan is reflected in development environment requirements that are documented in the RAD.

Transition environment requirements are requirements on the system used for the Transition to Operations phase of the STAR EPL. This phase consists of the installation of the integrated, tested pre-operational product processing system in the operational environment. A project may choose to transition the system from the development environment directly to the operational environment. In that case, the requirements should explicitly state this intention and should identify the specific operational environment. Alternatively, a project may choose to use a transition environment that is separate from the operational environment as an environment for acceptance testing. This may be a pre-existing environment that has been established for this purpose. In that case, the requirements should explicitly identify that environment. If a new transition environment is to be established, the requirements should include specific hardware requirements (e.g. CPUs, platforms, location) for the new environment.

Operational environment requirements are requirements on the system used for the Operations and Maintenance phase of a project. This phase consists of routine operations, reactive maintenance of the operational code, and science maintenance of the operational product. Typically, the operational organization will be selected during project planning and will be documented in the DPP. A basic system requirement is therefore the identification of this organization. It is critical that the characteristics of the selected operational environment will enable the operational product to meet its requirements (e.g., latency). The ability of the operational environment to meet operational product requirements may change as the product processing system design and code are developed. It is therefore critical that this ability be tracked as part of the project's Verification and Validation Plan (VVP).

Test environment requirements apply to the environment in which pre-operational code unit tests and system integration tests will be performed, including the test data that will be used. Requirements on the environment in which operational acceptance testing will be performed are addressed as transition environment requirements. A project may use the development environment to test the pre-operational system or it may choose to establish a separate test environment, Project constraints will usually determine this choice. For example, operations may request that the test environment be a clone of the operational environment, but cost factors may exclude establishing the development environment as an operational clone. In that case, the best solution may be to use a small operational clone environment as a separate test environment. A project may also choose to perform its pre-operational code unit tests in the development environment and then perform its system integration tests in an operational clone environment. In any case, these choices should be explicitly stated as requirements for the test environment.

Test environment requirements are not test plans. The VVP, Unit Test Plan (UTP) and System Test Plan (STP) describe how tests will be performed, and depend for their details on the characteristics of the test environment. Therefore, test environment requirements and test plans should be developed concurrently, with communication between the developers.

Although test data requirements are commonly considered to be system requirements, there are cases where they will be product requirements. These cases occur when the test data will be delivered to end users. In these cases, the statement of requirements in the RAD should explicitly identify the test data sets that will be delivered to end users.

Delivery requirements are requirements on the delivery of the operational product to end users. These include specific identification of end users, how often and by what means the deliveries are to be made, and product acceptance requirements. Clearly, the developers of delivery requirements must be in communication with the approved end users to ensure that the delivery requirements are acceptable to both the product producers and the end users.

Monitoring and Maintenance requirements address the routine monitoring of operations, reactive maintenance of operational code and the science maintenance of operational products. A project may adopt the STAR standard set of maintenance processes. In that case, it is sufficient to state this as the sole project maintenance requirement. If a project chooses to tailor its maintenance process, the requirements must state how this process will be tailored.

For purposes of analysis (c.f. Section 5 of this TD), system requirements should be characterized as either operational requirements or functional requirements. Operational requirements state “under what conditions” a function must be available or performable and address how the product will serve the users. Functional requirements address what the product must do to satisfy the operational requirements and define the necessary tasks, actions, or activities that must be accomplished.

4.6. Derived Requirements

Derived requirements are those requirements that are not directly traceable to a customer/user need or expectation, or a NESDIS mission goal, but instead are directly traceable to a basic requirement or to another derived requirement. Figure 4.3 illustrates the relation between basic requirements and derived requirements:

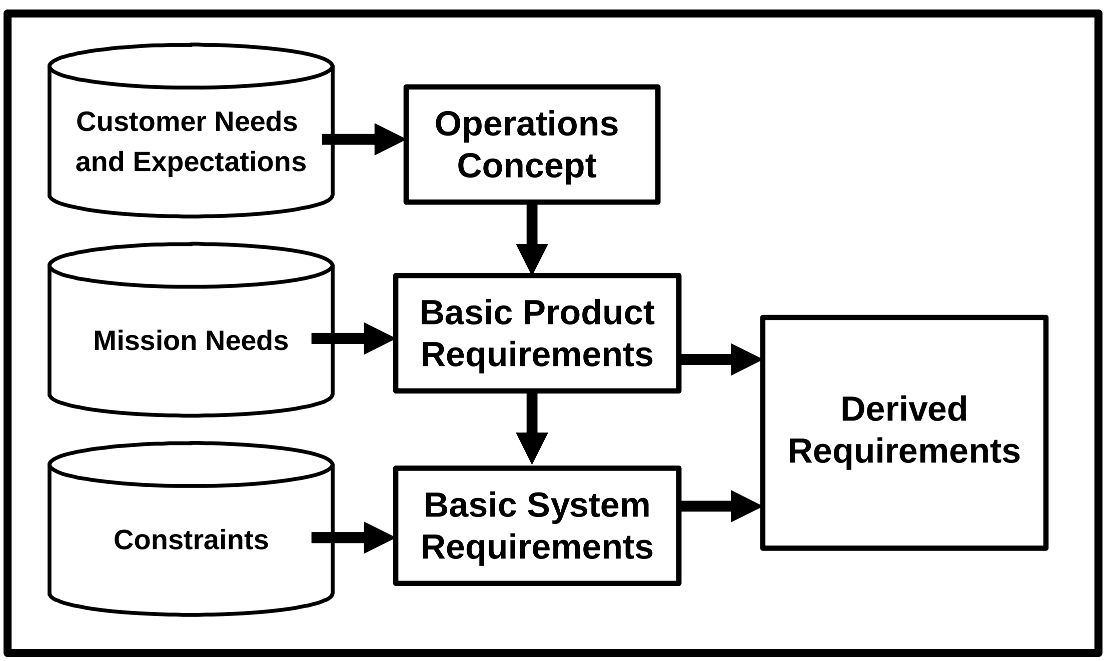

Figure 4.3 – Basic and Derived Requirements

Derived requirements are typically determined by analysis of basic requirements (c.f. Section 5 of this TD).

Derived requirements traceable to a basic product requirement are derived product requirements.

Derived product requirements address the cost and performance of other life-cycle phases (e.g., production, operations, and disposal) to the extent compatible with business objectives.

Derived requirements traceable to a basic system requirement are derived system requirements.

The structure of derived requirements may include multiple levels. That is, a derived requirement may be directly traceable to a basic requirement or another derived requirement. This trace should be documented in the RAD (c.f. Section 4.7 of this TD). The traceability of derived requirements is accomplished simply and automatically by the use of the standard numbering convention for project requirements.

-

- **Develop derived requirements from the basic requirements. Document in the RAD for review at PRR, PDR, and CDR, following guidelines in DG-6.2.**

-

Each requirement on the list of identified requirements should have a unique numerical identification. Basic requirements should be numbered 1.0, 2.0, 3.0, etc. Requirements derived from basic requirement 1.0 should be numbered 1.1, 1.2, 1.3, etc. Requirements derived from derived requirement 1.3 should be numbered 1.3.1, 1.3.2, 1.3.3, etc. Figure 4.4 illustrates this.

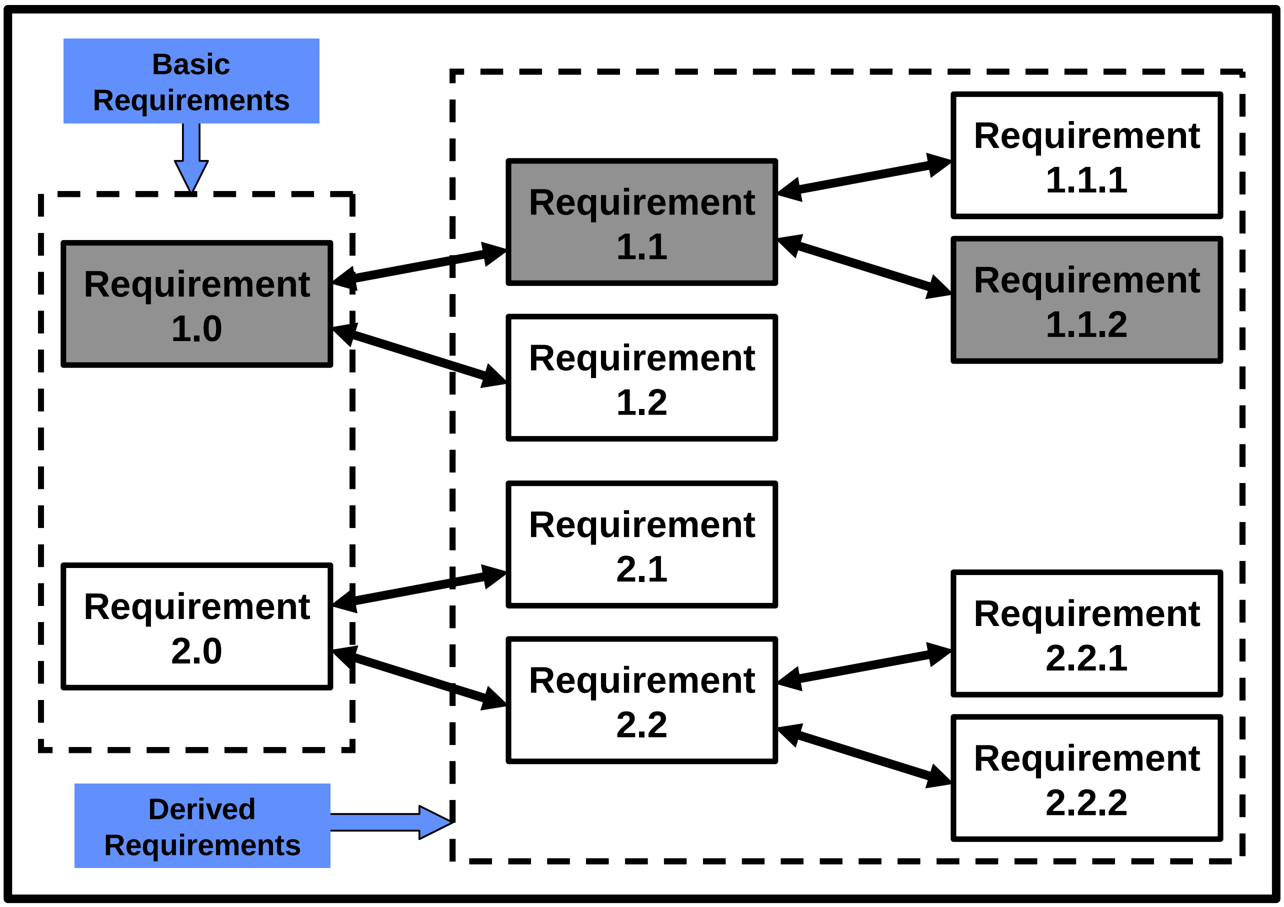

Figure 4.4 – Numbering Convention for Requirements

The two-sided arrows in Figure 4.4 illustrate the two-way traceability between higher and lower levels of requirements. The shaded requirements illustrate a particular example of traceability from requirement 1.1.2 to requirement 1.1 to requirement 1.0. Not shown in this figure, but implied, is the potential for lower levels of derived requirements (e.g. 1.1.2.1, etc.).

Any constraints on the purchase of equipment, software, or other resources should be reflected in explicit derived requirements. For example, if the customer will not accept COTS components in the delivered product, there should be an explicit derived requirement that COTS shall not be used in product components.

4.7. Requirements Documentation

The results from Requirements Identification should be documented in Section 2.3 of the RAD. The RAD is the core requirements document. It is developed iteratively throughout the Design phase and is maintained afterwards through the Build phase. Refer to DG-6.2 for RAD guidelines.

It is recommended that Section 2.3 of the RAD includes a summary of the requirements identification in a Requirements/Needs Matrix (RNM). The RNM links each basic requirement to a specific customer/user need or expectation that should be documented in the DPP and the OCD. Figure 4.5 shows an example.

		Requirements Driver			
Requirement Number	**Requirement Statement**	**Customer requests that an ATBD revision be reviewed at the PRR.**	**Customer needs an atmospheric vertical temperature profile**	**Customer expects the atmospheric vertical temperature profile to contain at least 32 layers**	**Customer needs a temperature accuracy of 0.5 K for each vertical layer**
0.0	**The STAR organization's set of standard processes shall be followed, except as specified in requirement 0.1.**	**DPP Section 5.7**			
0.1	**The exit criteria for the Project Requirements Review shall be the tailored exit criteria that are stated in the DPP.**	**DPP Section 5.7**			
1.0	**The product shall include an atmospheric vertical temperature profile**		**OCD Section 2.4.1**		
2.0	**The atmospheric vertical temperature profile shall contain at least 32 layers**			**OCD Section 2.4.2**	
3.0	**The reported temperature for each vertical layer shall have an accuracy of 0.5 K**				**OCD Section 2.4.2**

Figure 4.5 – Example of a Requirements/Needs Matrix

As Figure 4.5 shows, the RNM should contain requirements in its rows and requirements drivers in its columns. A cell is filled in if the requirement for its row is traceable to the driver for its column.

All basic requirements should be included. Derived requirements are included only if they relate directly to a requirements driver that has been documented in the DPP or the OCD.

The requirements drivers should be based on customer needs or expectations, the NESDIS strategic and mission plan, and tailored STAR EPL processes. These should be documented in the DPP or the OCD. If a requirements driver is not documented, the requirements traceable to that driver should not be approved.

To assist requirements reviewers, it is recommended that filled cells include a reference to the document section where the requirements driver is documented. It is also helpful to use fill colors to enhance readability of the RNM. Figure 4.5 illustrates this. It may not seem to be that helpful, since the example RNM is small, but typical RNMs will be much larger.

It is recommended that the RNM and all other requirements matrices be created in a Microsoft Excel file, with each matrix contained on its own sheet. The Excel file should be included as an Appendix to the RAD. A sheet can be pasted into the RAD as a Microsoft Word Picture, if it is sufficiently small to make a legible Picture.

At the conclusion of the requirements identification process, the requirements identifiers should produce a list of identified requirements. This list is an interim document. Its purpose is to facilitate requirements analysis (c.f. Section 5 of this TD). Eventually, this list will be superseded by the RAD (and often will form the basis for the initial draft of the RAD). A project may choose to use a draft version of RAD v1r0 instead of a separate interim list, with the understanding that a completed RAD v1r0 must include sections for Requirements Analysis, Requirements Allocation and Requirements Quality Assurance (c.f. DG-6.2).

- **Document the results from Requirements Identification in Section 2.3 of the RAD, following guidelines in DG-6.2. Include a Requirements/Needs Matrix (RNM) as an Appendix to the RAD. Produce a list of the identified requirements for use in requirements analysis.**

---

5. REQUIREMENTS ANALYSIS

Requirements analysis is the process of analyzing the identified requirements in light of the customer's needs, mission objectives, system constraints, and design constraints to develop more specific product, system, and process requirements for the system. In particular, many derived requirements will be developed from analysis of basic requirements.

Requirements analysis provides the basis for the approval, allocation and validation of requirements. Requirements analysis follows the identification of requirements. Requirements identifiers should provide a list of identified requirements (c.f. Section 4.7 of this TD) to the analysts. Requirements analysts should work from this list and other relevant project artifacts (e.g. DPP, OCD).

Requirements are analyzed to ensure that the established criteria are met. Conduct analyses of the requirements with the requirements provider(s) to ensure that a compatible, shared understanding is reached on the meaning of the requirements so the project participants can commit to them.

Requirements analysis includes:

- Acceptance analysis
- Technical analysis
- Quantitative analysis
- Functional analysis
- Prioritization

Most requirements analysis will be performed after the project plan has been established during the Plan phase and approved at the Gate 3 Review. However, some requirements analysis is necessary as a pre-condition for Gate 2 and Gate 3 approvals. This consists primarily of acceptance analysis of customer requirements with respect to the NESDIS mission and strategic plan (c.f. Section 5.1 of this TD).

5.1. Acceptance Analysis

Before requirements are accepted for analysis and disposition, they should pass standard quality criteria. Acceptance criteria include the following:

- Requirements are clearly and properly stated. If the requirements analyst is not sure that he understands the requirement statement, he should obtain clarification from the requirements identifier.
- Requirements are complete. Refer to the customer’s concept of operations and the relevant project artifacts (OCD, DPP) for evidence that a customer need or project goal has not been properly identified.
- Requirements are consistent with the NESDIS mission and strategic plan. This analysis should begin with the review of the project proposal at Gate 2. A Gate 2 review is limited to analysis of the customer needs and product descriptions as stated in the proposal. As the project requirements are more fully developed, it is necessary to ensure that they are not diverging from the NESDIS mission and strategic plan. The project manager and Development Lead should ensure that project requirements developers either have access to this plan or are provided with guidelines derived from this plan.
- Requirements are internally consistent with each other. This analysis should focus on obvious discrepancies between requirements statements. A more detailed analysis of requirements consistency will be made as part of the technical and functional analyses to follow.
- Requirements are uniquely identified.
- Requirements are traceable to their sources
- Requirements are completely traceable to higher level requirements

----

- ➢ **Perform an acceptance analysis of the requirements, using standard acceptance quality criteria. Requirements should be clearly and properly stated, complete with respect to customer needs and project goals, consistent with the NESDIS strategic and mission plan, internally consistent with each other, uniquely identified, traceable to their sources, and completely traceable to higher level requirements.**

-

5.2. Technical Analysis

The purpose of technical analysis is to develop product requirements in technical terms necessary for verification and for product and product-component design.

The customer requirements may be expressed in the customer's terms and may be non-technical descriptions. The product requirements are the expression of these requirements in technical terms that can be used for design decisions.

Analyze requirements to ensure that they are complete, feasible, realizable, and verifiable. While design determines the feasibility of a particular solution, technical requirements analysis addresses knowing which requirements affect feasibility. Identify key requirements that have a strong influence on cost, schedule, functionality, risk, or performance. Identify technical performance measures that will be tracked during the development effort.

Analyzing customer requirements requires an in-depth understanding of not just the customer requirements, but also the capabilities and limitations of hardware and software from which the product will be developed.

Analyze basic requirements to determine derived requirements that are needed to satisfy the objectives of higher level requirements. In light of the operational concept and scenarios (c.f. Section 4.3 of this TD), the requirements for one level of the product hierarchy should be analyzed to determine whether they are sufficient to meet the objectives of customers, other approved requirements sources, and the NESDIS strategic and mission plan. Identified gaps should drive additional basic requirements and lower level (derived) requirements.

In turn, lower level requirements should be analyzed to determine whether they are sufficient to meet the objectives of the higher level requirement. The analyzed requirements then provide the basis for more detailed and precise requirements for lower levels of the requirements hierarchy.

As requirements are defined and refined through technical analysis, their relationship to higher level requirements and the higher level defined functionality (c.f. Section 5.4 of this TD) must be understood to ensure satisfactory requirements verification. Requirements traceability matrices should be constructed to document the results of the technical requirements analysis. These matrices will be used to facilitate requirements quality assurance (c.f. Sections 7.1 and 7.4 of this TD).

Figure 5.1 illustrates the process flow for technical requirements analysis.

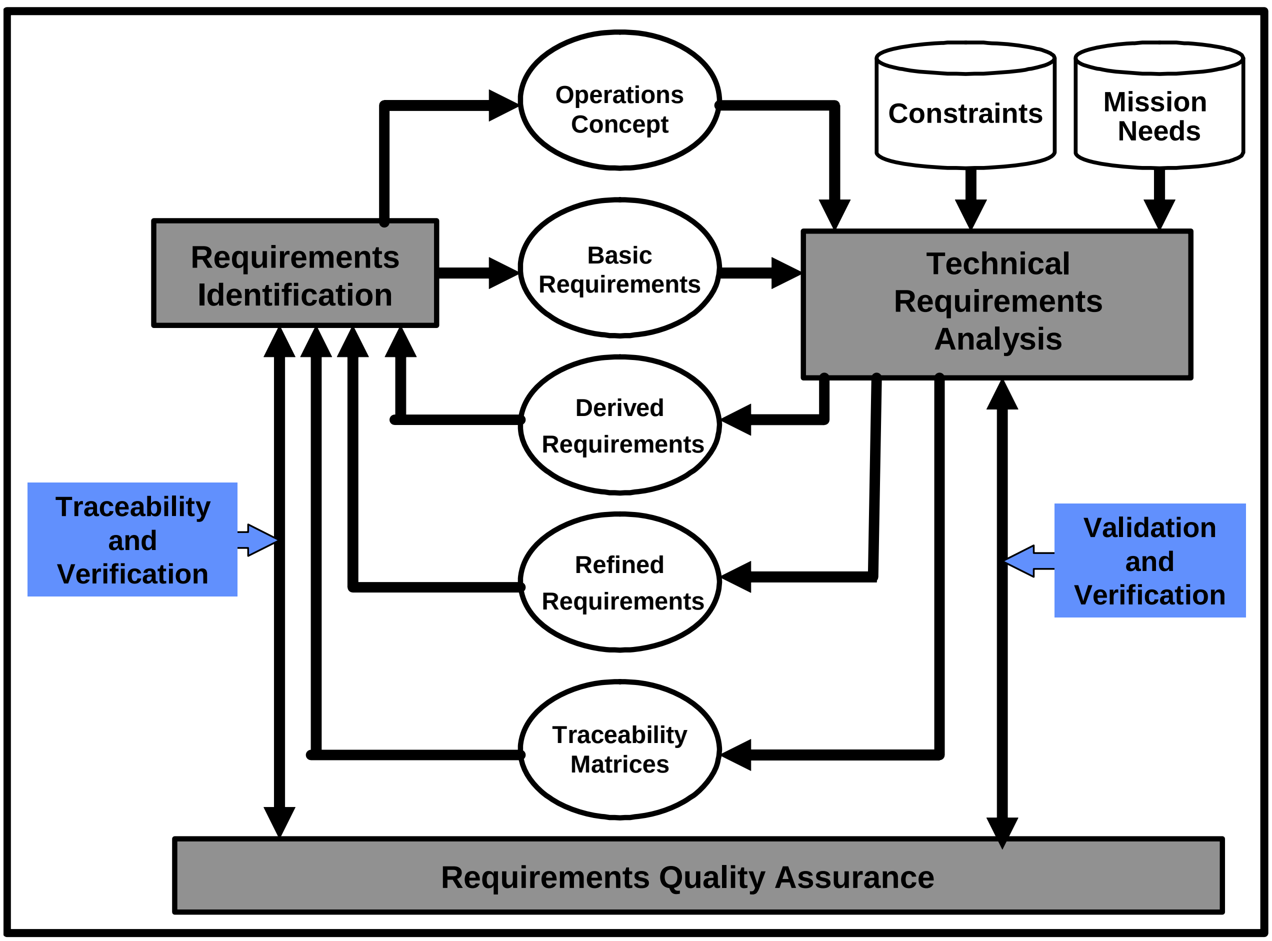

Figure 5.1 – Technical Requirements Analysis

Note that the process flow illustrated in Figure 5.1 is a subset of the process flow illustrated in Figure 3.2.

Figure 5.1 shows that requirements technical analysis is closely linked with requirements identification. The loops between the two are intended to be iterative, with analysis refining identification, until a satisfactory convergence is reached on a set of requirements that balances customer needs and NESDIS mission needs with constraints, including the capabilities and limitations of hardware and software from which the product will be developed.

Note that the operations concept (c.f. Section 4.3 of this TD) is included in a loop between Requirements Identification and Technical Requirements Analysis. Recall that the operations concept serves as a bridge between the requirements sources and the requirements. As the requirements are further identified and refined by technical analysis, the operations concept must also be refined to maintain the integrity of this bridge.

-

- **Perform a technical requirements analysis to develop derived requirements and requirements traceability, refine requirements, and refine the operations concept.**

-

5.3. Quantitative Analysis

Quantitative analysis is a specialized subset of technical analysis that is focused on performance requirements. Performance requirements must be specific and quantitative. Analysis should strike a balance between customer needs and expectations, whether quantitative or qualitative, and anticipated constraints. Consider cost, schedule and technical constraints. Consider the importance of the product performance to the NESDIS strategic and mission plan.

Figure 5.2 illustrates the process flow for quantitative requirements analysis.

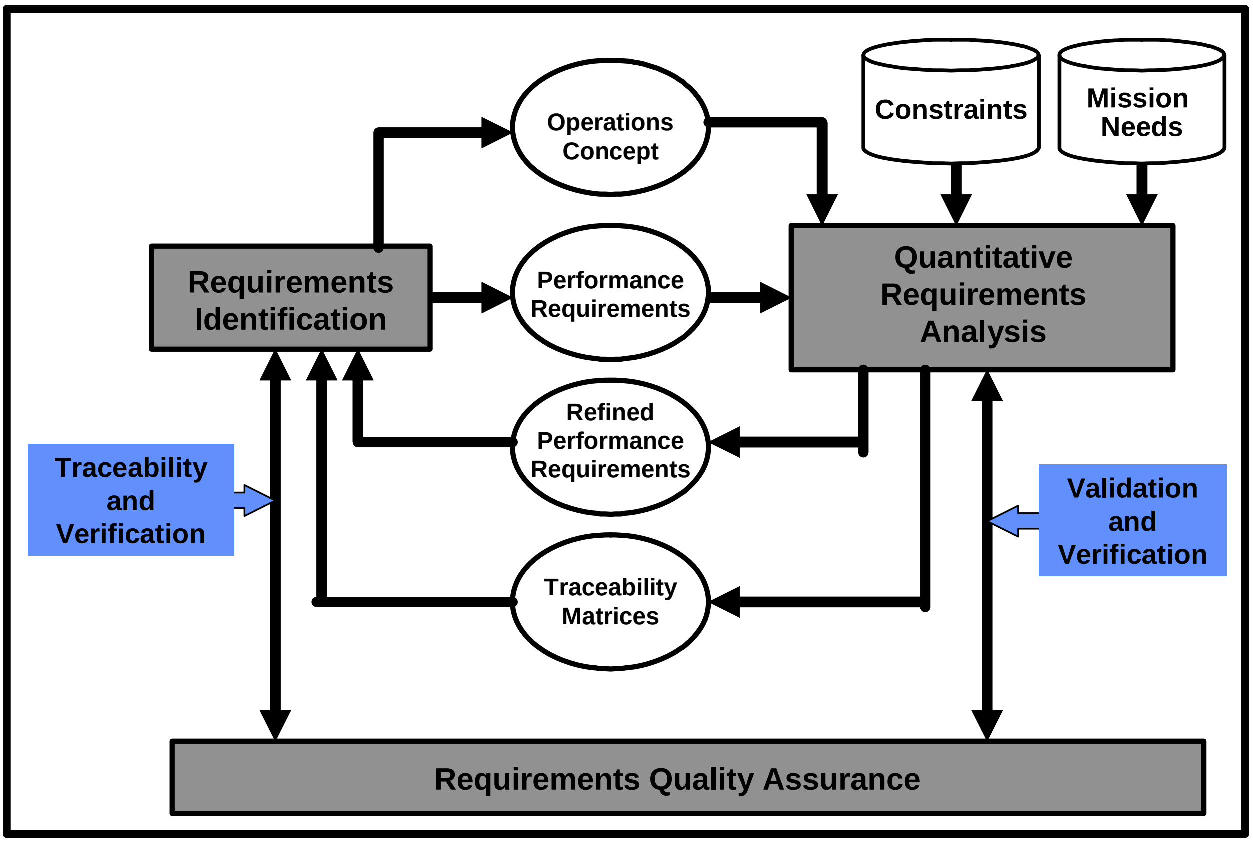

Figure 5.2 – Quantitative Requirements Analysis

Note the similarity of Fig 5.2 to Fig 5.1. This is expected, since quantitative analysis is a specialized technical analysis.

-

- **Perform a quantitative requirements analysis to develop derived product performance requirements and requirements traceability, refine product performance requirements, and refine the operations concept.**

-

Quantitative analysis of performance requirements may require testing of the performance of solutions, and therefore may need to be extended into the Build phase (steps 9-11) of the STAR EPL. In that case, the versions of the RAD developed during the Design phase (RAD v1r0 and its revisions) should explicitly state that the quantitative analysis of the performance requirements is provisional. This provisional status should be noted as a project risk that will require careful monitoring as coding and testing proceed.

5.4. Functional Analysis

Functional analysis is an analysis of requirements and operations concept to determine what a product processing system is intended to do. The purpose is to identify, describe, and relate the functions a system (or subsystem) must perform. The definition of functionality can include actions, sequence, inputs, outputs, or other information that communicates the manner in which a product will be produced and used. This is needed to allow for an effective allocation of requirements to product components and system components (c.f. Section 6.3 of this TD).

The standard steps of a functional analysis include:

1. **Draw System Diagram.** Draw a diagram with the system shown as a big circle at the center and show all the "outside" interfaces and influences the system must be concerned with. This helps to put the system in perspective and define what inter-actions are necessary to perform the job.
2. **List System Functions and Constraints.** The functions detail the transformation of system inputs into usable system outputs. Note that if a perceived "system output" is not provided to an outside entity, it is not truly an output. Sometimes outputs are erroneously specified because of a preconceived bias for a particular implementation. It is true that certain implementations would provide additional benefits to the user that are not specified by the user, but those issues are best left for the synthesis and tradeoff analyses tasks. Likewise, if a perceived "system input" does not contribute in some manner to the system output, then it is not truly a system input. The system constraints specify under what conditions the system must perform, such as, environmental requirements or theater of operations
3. **Specify External Interfaces.** Required interfaces should be thoroughly documented or referenced. These will represent outside constraints for your system. Writing a draft system architecture is a good place to start. Define, to the extent known, how to

operate with the outside entity with details, such as pinouts, timing, flow control, or data flows. Concentrate on those interfaces that are required by the user. Other outside interfaces may be needed depending on the system implementation, but you should wait on specifying those until the functional architecture has been developed and the synthesis activity is under way. Note that many times, people interaction is a critical interface for the system. In the early phases of the project, only the "types" of information may be given. During development activities, rapid prototyping is a good tool for nailing down the user's requirements and getting usable customer feedback. If the user has specified operator requirements, then the operators are an outside interface. If not, then the operator interface should be considered an internal interface and not addressed in this step.

4. **Partition into Functional Groupings.** This is the step where you break down that big system circle from step 1 into the next layer of detail and show the internal interfaces between them. Analyze requirements to identify logical or functional partitions (e.g., sub-functions). These sub-functions will be used to allocate requirements to functional partitions (c.f. Section 6.1 of this TD). Partition requirements into groups, based on established criteria (e.g., similar functionality, performance, or coupling). Consider the sequencing of time-critical functions. There is no "right" way to partition the system functional requirements into logical groupings, so this becomes more art than science. Some of the techniques that can be used are to form groupings based on sequence of operation, interfaces required, common procedures, or logical similarity. One goal is to minimize the amount of interaction between the functional groupings.

5. **Iterate As Necessary.** Each of your functional groupings can now be viewed as a subsystem. You have inputs and outputs (some internal to system, some external), and requirements for each function.

Figure 5.3 illustrates the standard functional analysis steps.

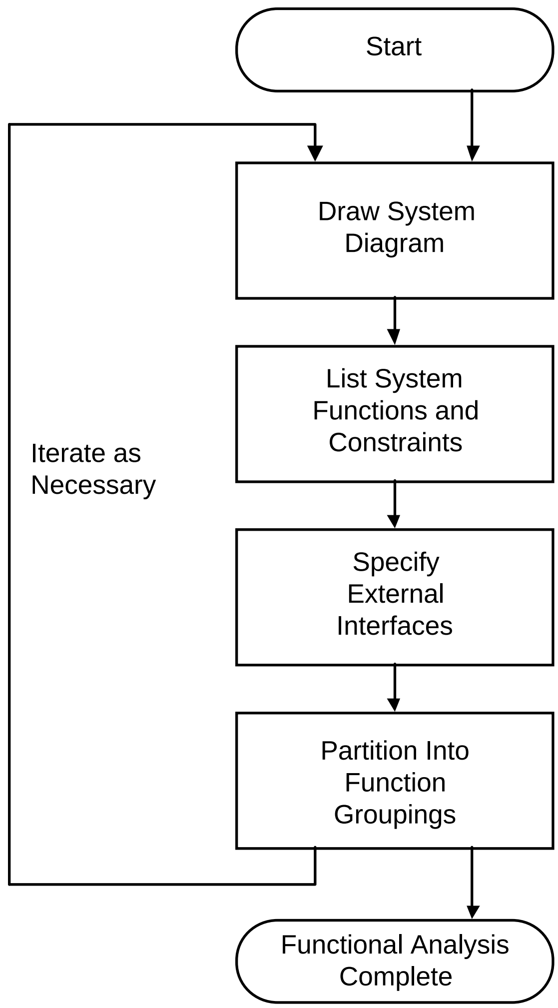

Figure 5.3 – Standard Functional Analysis Steps

Reiterate these steps until you reach the level of detail necessary. What is the level of detail necessary? Only perform the amount sufficient to support the corresponding level of synthesis. Generally, this depends on the system problem, the phase of the project, and how much of the development is new. During the early phases of the project, the level of detail is minimized because the focus is on understanding the interrelationships with the overall mission objective and in considering the maximum number of alternatives for satisfying the objectives. As the project progresses, the level of detail is increased because the synthesis task needs greater amounts of detail.

For PRR, it is expected that functional analysis will result in a decomposition of basic requirements into derived functional requirements in sufficient detail so that preliminary design solutions can be synthesized. Functional requirements describe “what” the system must do independent of the physical or actual implementation. It is important to maintain this independence in order to objectively evaluate alternative solutions during synthesis. Without proper care, implementation biases at this stage may force you into specifying a non-optimal solution.

The definition of functions, their logical groupings, and their association with requirements is referred to as a functional architecture. Functional architecture is the hierarchical arrangement of functions, their internal and external (external to the aggregation itself) functional interfaces and external physical interfaces, their respective functional and performance requirements, and their design constraints. Functional architecture serves as the bridge between the operations concept (c.f. Section 4.3 of this TD) and the system architecture of design components.

In developing a functional architecture, it is recommended that functions be numbered for identification, in a fashion similar to the numbering of requirements (c.f. Section 4.6 of this TD). This will help in the allocation of requirements to functions (c.f. Section 6.1 of this TD) and in the traceability of requirements (c.f. Section 7.1.2 of this TD).

Note that functional analysis does not presume that the resulting functional architecture must be implemented with a functionally oriented software design.

Alternate concepts are developed by defining and allocating functions, then synthesizing system architecture. Performance of alternate architectures are compared, and the best solution chosen. This is an essential part of the Preliminary Design step of the STAR EPL.

The process iterates as the design progresses (and frequently, as the customer’s requirements change) and more information becomes available until the system is complete.

Figure 5.4 illustrates the process flow for functional requirements analysis.

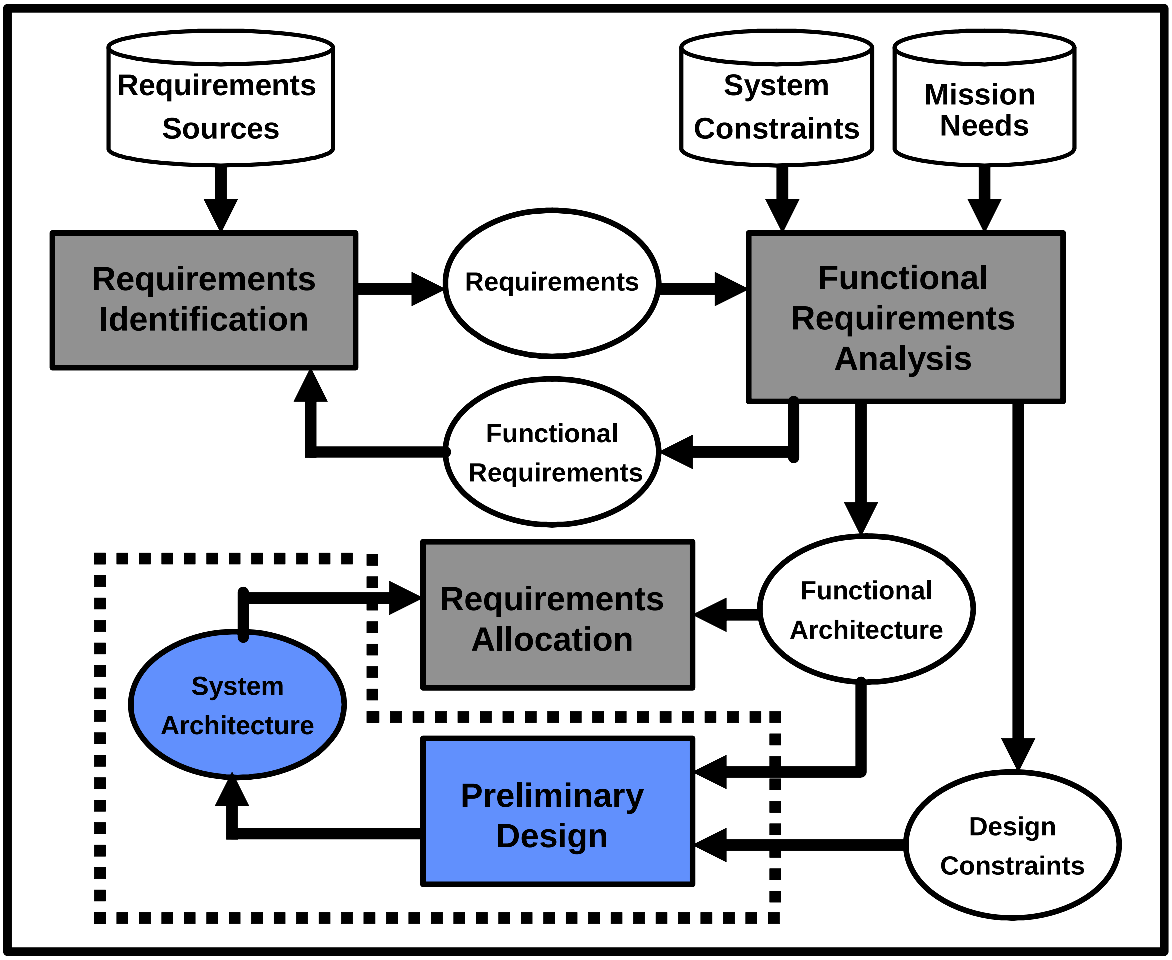

Figure 5.4 – Functional Requirements Analysis

Functional analysis of requirements, allocation of requirements and design (system architecture) are closely related. Note that Figure 5.4 includes some process flow, inside the dotted border and shaded in light blue, that is actually part of the Preliminary Design process. Note also that this part of the Preliminary Design process feeds back into Requirements Allocation, through the system architecture input. In practice, these processes are iterative, as illustrated in Figure 5.4 (recall also Figure 3.3). Therefore, the status of requirements, requirements allocation, and design must be reviewed at all technical reviews of the Design phase (PRR, PDR and CDR) to ensure that requirements are properly translated into design solutions.

-

- **Perform a functional requirements analysis to develop derived system requirements and requirements traceability, develop a functional architecture for system design, and identify constraints on the design.**

-

5.5. Prioritization

If all requirements are treated equally, an endless loop of tradeoffs addressing conflicting requirements and design may ensue. A priority based on customer desires and the NESDIS strategic and mission plan must therefore be established.

Identify key requirements that have a strong influence on cost, schedule, functionality, risk, or performance.

-

- **Prioritize requirements based on customer desires and the NESDIS strategic and mission plan.**

-

The Development team should use proven models, simulations, and prototyping to analyze the balance of customer needs, other stakeholder needs and system constraints. Project management should oversee this analysis. Stakeholder needs and constraints can address cost, schedule, performance, functionality, reusable components, maintainability, or risk. Results of the analyses can be used to reduce the cost of the product and the risk in developing the product. Consultation with customers and end users is essential.

-

- **Analyze the balance of customer needs, other stakeholder needs, and system constraints to reduce the cost of the product and the risk in developing the product.**

-

Explore the adequacy and completeness of requirements by developing product representations (e.g., prototypes, simulations, models, scenarios, and storyboards) and by obtaining feedback about them from relevant stakeholders. This activity is a bridge between operations concept development and risk assessment, which should occur in parallel. Relations between requirements should be captured in a requirements matrix for consideration during change management and requirements allocation. Relationships include dependencies in which a change in one requirement may affect other requirements.

- **Explore the adequacy and completeness of requirements by developing product representations (e.g., prototypes, simulations, models, scenarios, and storyboards).**

- **Analyze the relations between requirements for consideration during requirements change management and allocation.**

5.6. Analysis Methods

Customer requirements arrive in many forms. They can be clearly stated, unambiguous, and non-conflicting or they can be vague, confusing, and conflicting. The complexity of the end product can also vary considerably, from a relatively simple component to an extremely complicated weapons system. However, the customer’s desires remain the same: development of an item that meets their performance, cost, schedule, and supportability needs. Critical to achieving this end product is a method or methods of requirement analysis and system concept definition.

Regardless of the complexity of the product, the basic steps in requirements analysis remain the same:

(1) Verify and understand customer needs and requirements

(2) Analyze these requirements to the extent needed to validate (or choose) and implement the proposed functional definition

(3) Determine the internal interface or connections among the various functions

(4) Define top-level key characteristics of the system which are essential to ensuring that a balanced and customer-focused approach is implemented.

The Quality Function Deployment (QFD) analysis method is a proven, reliable technique for translating customer needs into design requirements. QFD is described in Appendix A of this TD.

6. REQUIREMENTS ALLOCATION

The purpose of requirements allocation is to identify product and system components and trace each component to one or more requirement so that a system architecture that will meet all project requirements can be designed.

Figure 6.1 is a context diagram that shows the external flows into and out of the requirements allocation sub-process.

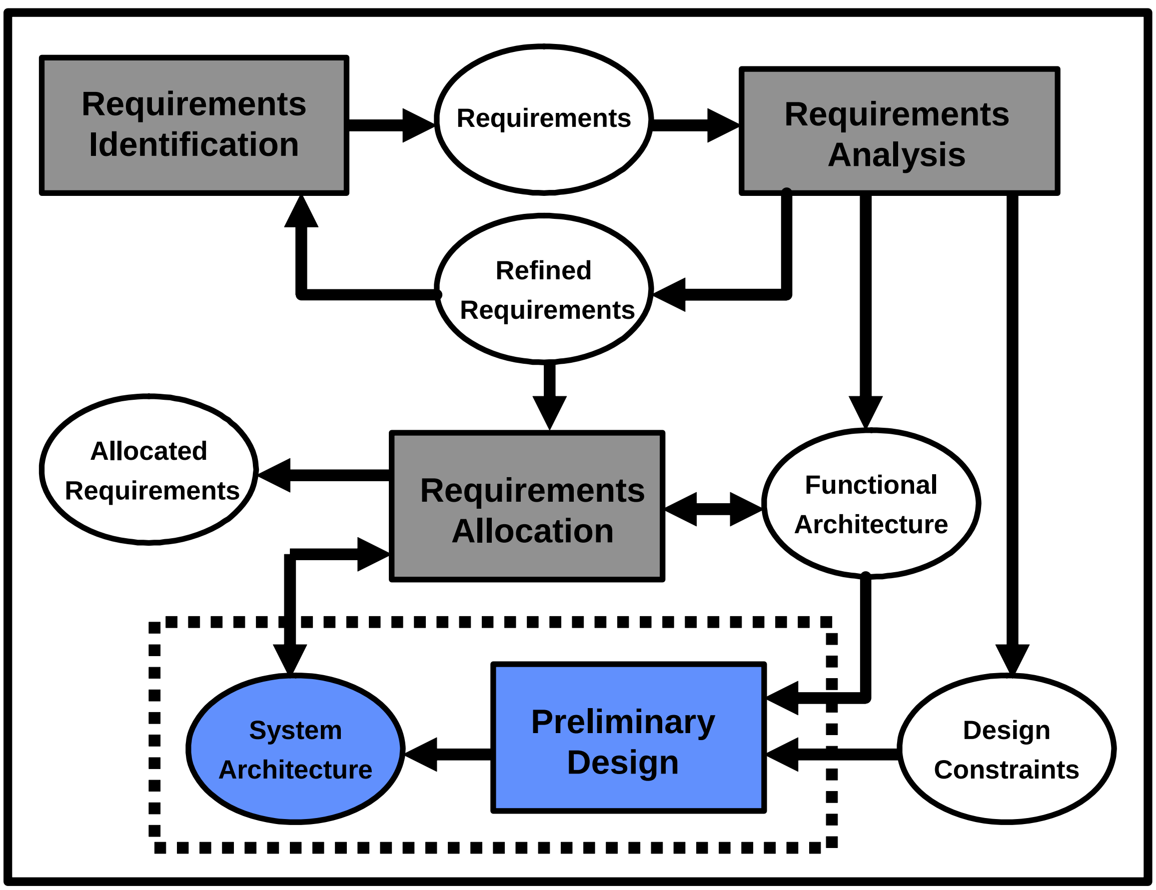

Figure 6.1 – Requirements Allocation Context

These flows will be discussed in Section 6.1, 6.2, and 6.3 of this TD.

Requirements allocation:

- Helps ensure that the finished system will meet all requirements
- Identifies where each portion of a requirement is to be verified
- Identifies the requirement tradeoffs that were made
- Provides the means to monitor and track progress against the user's requirements

Make a complete requirements allocation for each alternative approach. Establish the requirements associated with the selected set of alternatives as the set of allocated requirements to those product components. Selecting product components that best satisfy the criteria establishes the requirement allocations to product components. Lower level requirements are generated from the selected alternative and used to develop the product-component design. Interface requirements among product components are described, primarily functionally. Physical interface descriptions are included in the documentation for interfaces to items and activities external to the product.

-

➢ **Make a complete requirements allocation for each alternative approach. Establish the requirements associated with the selected set of alternatives as the set of allocated requirements to those product components.**

-

There may be cases where a project does not wish to analyze alternative solutions. In that case, the PRR should decide whether a trade study of alternative solutions should be conducted for PDR. If a project wishes to bypass the analysis of alternative solutions, it must provide a convincing rationale (examples: (1) strong algorithm heritage, (2) PUSH user algorithm is the project driver).

The requirements allocation sub-process consists of three sequential sub-processes:

1) Allocation of requirements to functions
2) Allocation of functions to operational modes
3) Allocation of requirements to system and product components

Allocation often includes allocation of errors to product and system components (i.e. error budgeting). In the STAR EPL process, error budgeting is usually deferred to the Build phase and is documented in the Verification and Validation Report (VVR) instead of the RAD. Therefore, error budgeting will not be discussed in this TD. Refer to TD-6 and DG-11.4 for details.

Figure 6.2 provides an illustration of the requirements allocation process flow, showing the three sub-processes.

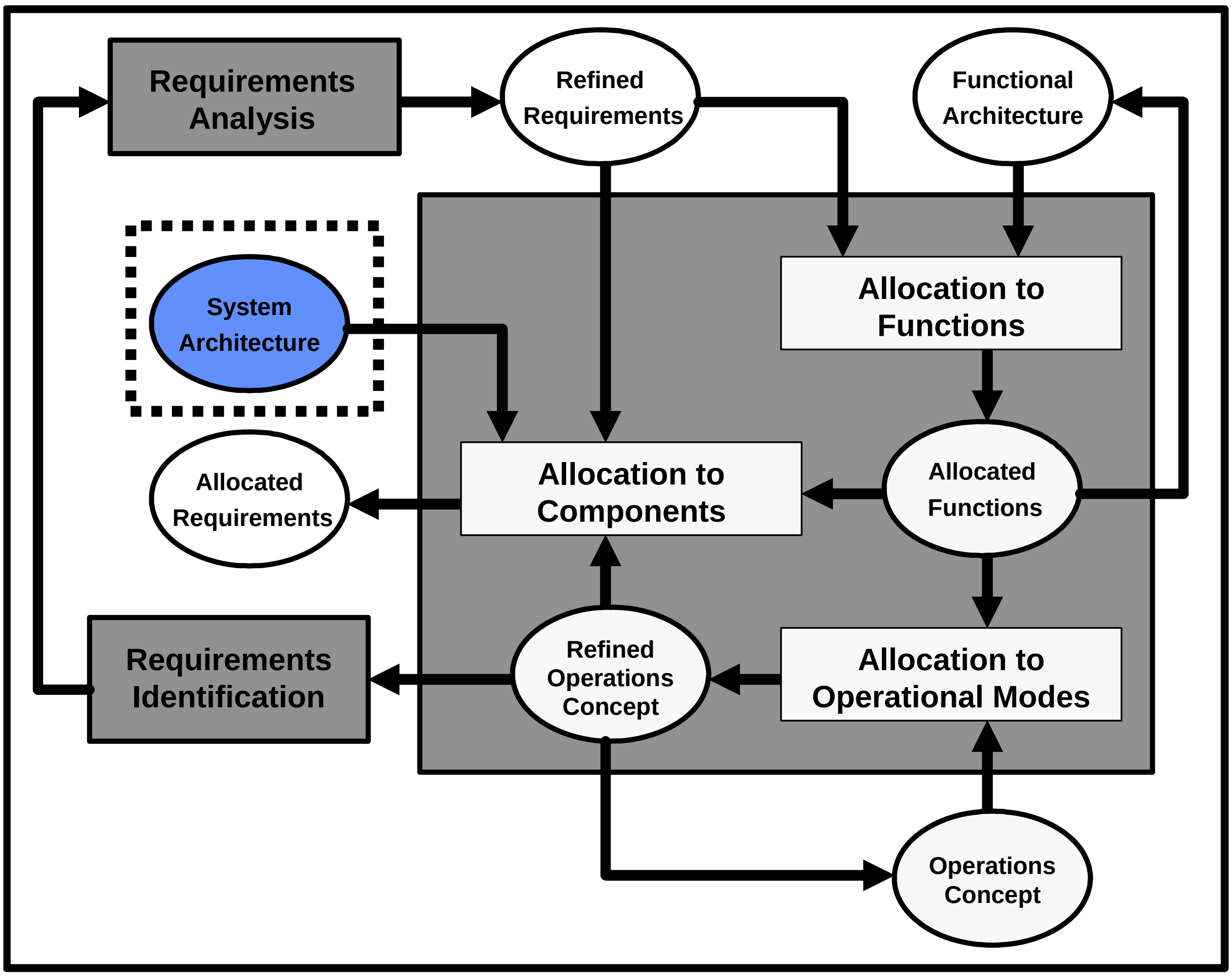

Figure 6.2 – Requirements Allocation Process Flow

Each of the three main requirements allocation steps is discussed in Sections 6.1, 6.2, and 6.3. Section 6.4 discusses the documentation of the requirements allocation.

6.1. Allocation to Functions

Figure 6.3 provides an illustration of the process flow for the allocation of requirements to functions sub-process, a subset of Figure 6.2.

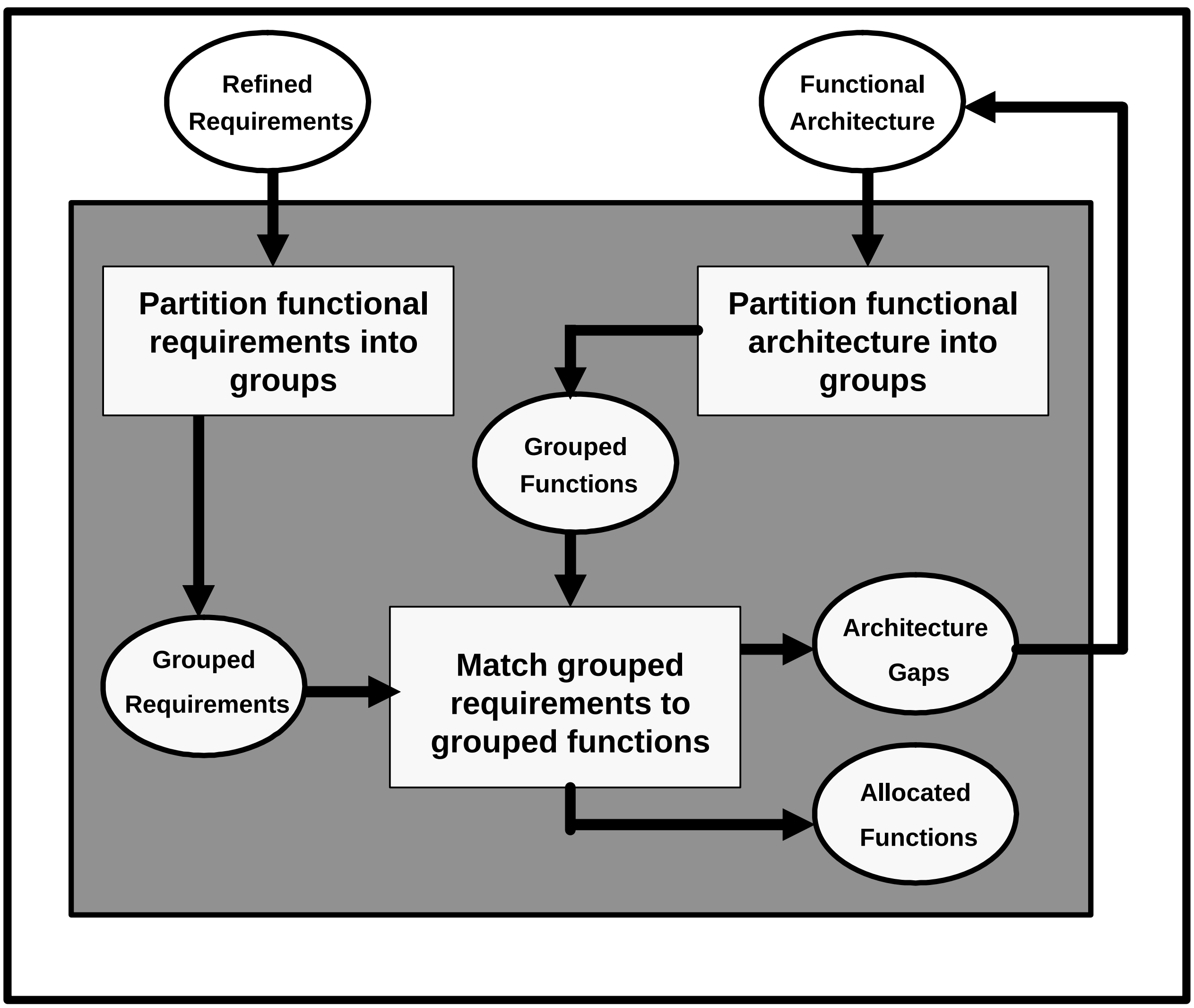

Figure 6.3 – Allocation to Functions Sub-Process Flow

Functional analysis of requirements (c.f. Section 5.4 of this TD) produces functional requirements traceable to product and system requirements. Functional requirements address what the product must do to satisfy the operational requirements and define the necessary tasks, actions, or activities that must be accomplished. Operational requirements state “under what conditions” a function must be available or performable and address how the product will serve the users.

Functional analysis of requirements also produces a functional architecture. Functional architecture is the hierarchical arrangement of functions and their sub-functions, the internal and external (external to the aggregation itself) functional interfaces and external physical interfaces, their respective functional and performance requirements, and their design constraints.

To allocate requirements to functions, partition functional requirements into groups, based on established criteria (e.g., similar functionality, performance, or coupling), that can be matched to corresponding groups, or partitions, of the functional architecture. There is no “right” way to partition the system functional requirements into logical groupings, so this becomes more art than science. Some of the techniques that can be used are to form groupings based on sequence of operation, interfaces required, common procedures, or logical similarity. One goal is to minimize the amount of interaction between the functional groupings.

- **Partition functional requirements into groups, based on established criteria (e.g., similar functionality, performance, or coupling), that can be matched to corresponding groups, or partitions, of the functional architecture.**

To facilitate the grouping of functions, it is useful to be able to identify the functions by a number that indicates their relationship to other functions. In that way, functions in the same group should have a numerical ID in common, in a way that is similar to the numbering of requirements (c.f. Section 4.6 of this TD). As functions are grouped, their numbering can be adjusted iteratively until the function numbers and the function groups are consistent.

The allocation of requirements to functions is complete when all functional requirements have been matched to a function or group of functions in the functional architecture. If you are having trouble finding a match for all functional requirements, it may be an indication that the functional architecture is incomplete and needs refinement. The feedback loop shown in Figure 6.3 (also in Figure 6.2) between the functional architecture and the

Allocation to Functions sub-process is intended to accommodate this possible need for iterative refinement of the functional architecture input to requirements allocation.

6.2. Allocation to Operational Modes

An operational mode is a system state that exhibits predictable behavior that will meet all operational requirements. An operational mode establishes conditions that dictate the functions required to perform desired actions that will enable the system to meet all operational requirements. It is therefore a bridge between operational requirements and functional requirements.

When the functional architecture, and the allocation of requirements to its functions, is complete, the functions should be allocated to operational modes. This is necessary to ensure that the functional requirements are sufficient to meet the operational requirements. All relevant operational modes should be obtainable from the operations concept (c.f. Section 4.3 of this TD).

-

- **Allocate functions to operational modes, when the functional architecture, and the allocation of requirements to its functions, is complete.**

-

The procedure for allocating functions to operational modes is similar to the procedure for allocating functional requirements to functions (c.f. Section 6.1 of this TD).

To allocate functions to operational modes, partition functions into groups, based on established criteria (e.g., similar functionality, performance, or coupling), that can be matched to corresponding groups, or sequences, of the operations concept. By matching the grouped operational sequences to the grouped functions, the functions are allocated to the operational modes.

-

- **Partition functions into groups, based on established criteria (e.g., similar functionality, performance, or coupling), that can be matched to corresponding groups, or sequences, of the operations concept.**

-

Figure 6.4 provides an illustration of the process flow for the allocation of requirements to functions, a sub-process of Figure 6.2.

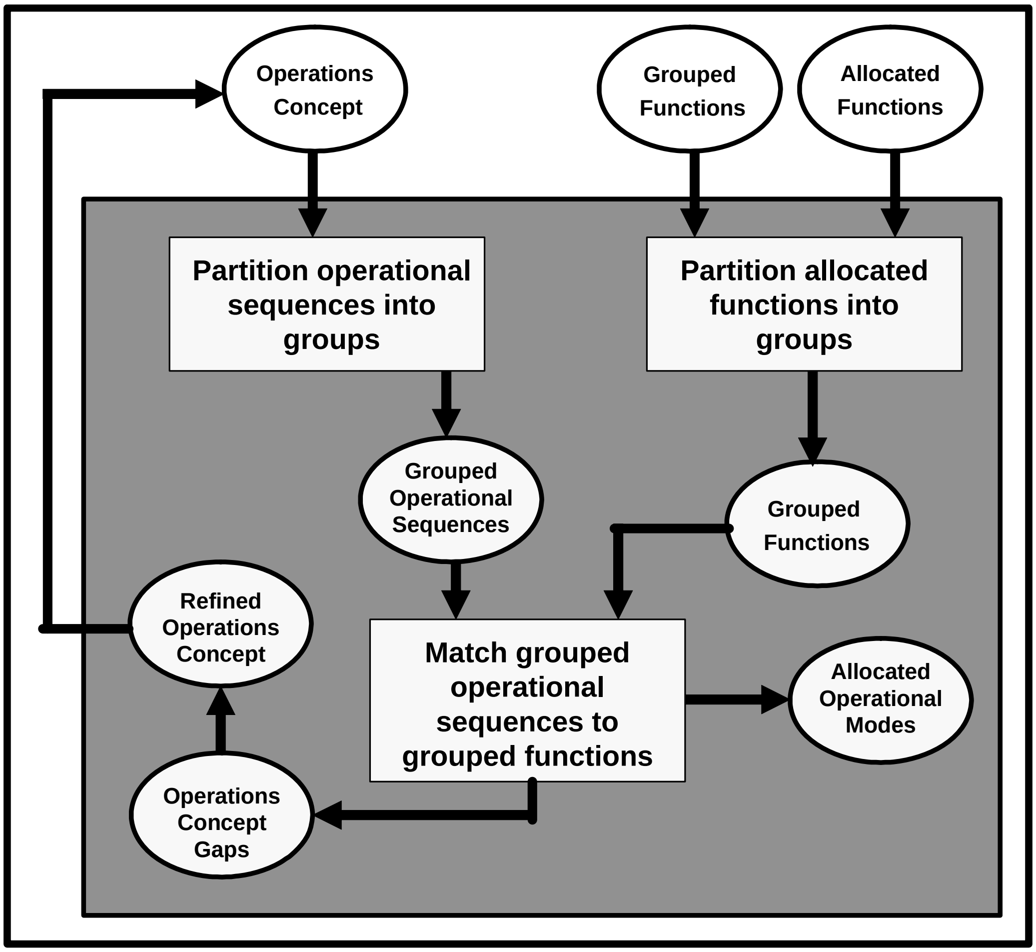

Figure 6.4 – Allocation to Operational Modes Sub-Process Flow

Start by attempting to match the grouped operational sequences to the function groups that were established in the Allocate to Functions sub-process (c.f. Section 6.1 of this TD) as a starting point. This function grouping should be a good first guess, since it was derived from the functional architecture that in turn is traceable to the operations concept itself. There may be some variation, however, because the requirements and functional architecture have been iterated since the original requirements identification was established from the operations concept. This variation may cause gaps between the current state of requirements allocation and the original operations concept.

The allocation of functions to operational modes is complete when all functions have been matched to an action in an operational sequence. If you are having trouble finding a match for all functions, try re-grouping the functions to better fit the operational sequences. If there are still functions that cannot be matched to the operations concept, it may be an indication that the operations concept is incomplete and needs refinement. The feedback loop shown in Figure 6.4 (also in Figure 6.2) between operations concept and the Allocation to Operational Modes sub-process is intended to accommodate this possible need for iterative refinement of the operations concept input to requirements identification, analysis, and allocation.

Continue to iterate operations concept, requirements identification, requirements analysis, and requirements allocation until all functions in the functional architecture are allocated to operational modes that are consistent with the operations concept. When this has been achieved, the project is ready for preliminary design. This is the expected status of the project at the PRR.

- **Continue to iterate operations concept, requirements identification, requirements analysis, and requirements allocation until all functions in the functional architecture are allocated to operational modes that are consistent with the operations concept.**

6.3. Allocation to Components

The allocation of requirements to system and product components of the product processing system is the final sequential sub-process of requirements Allocation. Here, the system architecture, refined operations concept, refined requirements and allocated functions are all used to match the requirements to the elements of the design solution (i.e. the product components and system components).

Product components are defined as any item that will be integrated to form the end-use product, i.e. these are the deliverable items. System components are defined as any item that is necessary or useful for building the end-use product, but will not be delivered to customers and/or end users.

Allocate requirements to product components that have been identified in the system architecture, as documented in the latest version of the SWA. The version 1 system architecture is developed prior to requirements development solely for the purpose of supporting research coding and typically will not be mature enough for a complete allocation. For PRR, it is sufficient to identify those product components in the architecture that are likely to be retained in the version 2 architecture, and allocate pertinent requirements to those components. As previously noted, requirements allocation will be developed iteratively with design development (version 2 system architecture).

Allocate requirements to system components that have been identified in the system architecture, as documented in the latest version of the SWA. The version 1 system architecture is developed prior to requirements development solely for the purpose of supporting research coding and typically will not include system components.

- **Allocate requirements to product components and system components that have been identified in the system architecture, as documented in the latest version of the SWA.**

The requirements for product components of the defined solution include allocation of product performance; design constraints; and fit, form, and function to meet requirements and facilitate production. In cases where a higher level requirement specifies performance that will be the responsibility of two or more product components, the performance must be partitioned for unique allocation to each product component as a derived requirement.

Allocate design constraints to product components. Design constraints include specifications on product components that are derived from design decisions, rather than higher level requirements. Design constraints that affect requirements can be addressed at PRR if they are understood that early in the process (version 1 architecture). More commonly, design constraints are addressed at PDR and CDR.

- **Allocate design constraints to product components.**

For each requirement, produce a list of each system or product component that is designed to satisfy the requirement. Document this list in the RAD. Explain how the set of allocated components working together will fully satisfy the requirement. Document this explanation in the RAD.

- **Produce a list of each system or product component that is designed to satisfy each requirement. Document this list in the RAD.**

- **Explain how the set of allocated components working together will fully satisfy the requirement. Document this explanation in the RAD.**

6.4. Allocation Documentation

This TD contains numerous references to the RAD. The documentation of requirements allocation provides traceability of design to requirements and credibility to the project verification and validation plan. The RAD is the core requirements document and, together with the DPP, can be considered as the most important project artifact. It is the RAD that ties all project tasks described in the DPP to the requirements that provide the rationale for these tasks.

The RAD should be established, developed and maintained by all project stakeholders tasked to develop project requirements. Refer to STAR EPL process asset DG-6.2 for RAD document guidelines. It is essential that all project stakeholders involved in developing the RAD be familiar with the guidelines in DG-6.2.

This TD has made numerous references to items that should be documented in the RAD. Project stakeholders involved in developing the RAD should ensure that these items are documented in the RAD, following the guidelines in DG-6.2.

Summarize the requirements allocations in a Requirements Allocation Sheet (RAS). The RAS can be created as a table or imported from a Microsoft Excel spreadsheet to a Microsoft Word Object.

-

- **Summarize the requirements allocations in a Requirements Allocation Sheet (RAS) in the RAD.**

-

The RAS is a matrix. The rows consist of the requirements, with one row for each requirement. It is recommended that the requirements be listed in numerical order, with derived requirements listed after their basic requirements, as follows:

Requirement 0.0

Requirement 0.1

Requirement 0.1.1

Requirement 0.1.2

Requirement 0.2

Requirement 0.2.1

…….

Requirement 1.0

Requirement 1.1

etc.

The columns consist of the product and system components of the system architecture. It is helpful to number the components. In fact, it is the standard practice to number each component in the system architecture. The component numbers can be obtained from the latest version of the SWA. The requirements developers should consult with the developers of the system architecture to ensure that the correct component numbers are used in the RAS~~D~~.

The RAS should be included as an Appendix document to the RAD. Figure 6.5 illustrates an example RAS.

	PRODUCT AND SYSTEM COMPONENTS						
Requirement Number	Check L1C	L1C Subsetter	Global Grids	Global Binaries	Global Matchups	Off-line components	Interaction with Stakeholders
1							
1.1							
1.2							
1.3							
1.4							
1.5							
1.5.1							
1.5.2							
1.5.3							
1.5.4							
1.5.5							
2							
2.1							
2.1.1							
2.2							
2.2.1							
3							
3.1							
4							
4.1							
4.2							
5							
5.1							
5.2							
5.3							
5.4							
5.5							
5.5.1							
5.5.2							
5.5.3							
5.5.4							
5.6							
5.7							
6							
6.1							
6.1.1							
7							
7.1							
7.2							
8							
8.1							

Figure 6.5 – An example of a Requirements Allocation Sheet

7. REQUIREMENTS QUALITY ASSURANCE

Requirements QA is an activity that oversees all of the requirements sub-processes. Its purposes are:

- to ensure that all process requirements, product requirements and system requirements are developed according to standards
- to ensure that all requirements are traceable to drivers and other requirements
- to ensure that the requirements and requirements allocation provide a satisfactory balance between customer/user needs and expectations, NESDIS mission goals, technical feasibility, the available resources and external constraints.
- to ensure that all requirements are verifiable

Requirements QA consists of requirements traceability, tracking, validation and verification.

7.1. Requirements Traceability

Every requirement exists because of a higher level driver. This driver is either a customer need or expectation, the NESDIS mission and strategic plan, or a higher level requirement. In that sense, every requirement is a derived requirement. Requirements traceability is the ability to relate each requirement to its higher level and lower level requirements. Thjs activity is essential for maintaining the integrity of the requirements as the project matures, because the requirements or the requirements drivers may change during the project's product lifecycle. When this occurs, all requirements that are traceable to the changed requirement or driver must be analyzed to determine whether they must be changed or possibly deleted. Traceability extends to all requirements, including functional requirements. Often, a change to the requirements or requirements drivers will necessitate a new functional analysis and a revised functional architecture. Therefore, the traceability of all requirements, including functional requirements, must be documented.

7.1.1. Vertical Traceability

Requirements Traceability includes traceability from a basic requirement to its driver and to its lower level derived requirements and from the lower level requirements back to their higher level sources. This traceability is called vertical traceability because it moves across levels. Vertical traceability of all requirements should be established for PRR and documented in RAD v1r0.

If the requirements are numbered according to the standard numbering convention (c.f. Section 4.6 and Figure 4.4 of this TD), vertical traceability is a straightforward combination of the RNM (relating basic requirements to their drivers, c.f. Section 4.7 of this TD) and the requirements numbers (relating each requirement to its higher and lower level requirements). That is, each basic requirement can be traced to its driver through the RNM and each derived requirement can be traced to higher level requirements that contain the same higher level number (e.g. Requirement 3.2.7.5 can be traced to Requirements 3.2.7, 3.2, 3.0 and the driver of 3.0).

It is recommended that vertical traceability be documented in the RAD by a Requirements Vertical Traceability Matrix (RVTM) that summarizes the two-way vertical traceability from each requirements driver to its basic requirement and its derived requirements.

-

- **Document vertical traceability in the RAD by a Requirements Vertical Traceability Matrix (RVTM) that summarizes the two-way vertical traceability from each requirements driver to its basic requirement and its derived requirements.**

-

The rows of the RVTM contain each requirements driver (obtained from the columns of the RNM, (c.f. Section 4.7 of this TD) and each requirement (obtained from the rows of the RAS, c.f. Section 6.4 of this TD). It is recommended that the drivers be listed first, followed by the requirements in numerical order as listed in the RAS.

The columns of the RVTM should be identical in content and order to the rows, thereby creating a square symmetric matrix whose diagonal elements are identities.

Each cell in the RVTM consists of a binary condition (e.g., yes or no, 0 or 1, filled or blank) that indicates whether the requirement of its column is traceable to the requirement of its row.

An example RVTM is shown as Figure 7.1.

		Requirement Number														
Requirement Number	**Requirement Statement**	**D1**	**D2**	**D3**	**D4**	**0.0**	**0.1**	**1.0**	**1.1**	**1.1.1**	**2.0**	**2.1**	**2.1.1**	**2.1.1.1**	**3.0**	**3.1**
D1	Customer requests that an ATBD revision be reviewed at the PRR.															
D2	Customer needs an atmospheric vertical temperature profile															
D3	Customer expects the atmospheric vertical temperature profile to contain at least 32 layers															
D4	Customer needs a temperature accuracy of 0.5 K for each vertical layer															
0.0	The STAR organization's set of standard processes shall be followed, except as specified in requirement 0.1.															
0.1	The exit criteria for the Project Requirements Review shall be the tailored exit criteria that are stated in the DPP.															
1.0	The product shall include an atmospheric vertical temperature profile															
1.1	The product processing system shall calculate atmospherical vertical temperature profiles from infrared sounder data															
1.1.1	There shall be a code module that reads infrared sounder data and calculates atmospheric vertical temperature profiles															
2.0	The atmospheric vertical temperature profile shall contain at least 32 layers															
2.1	There shall be a code module that reads infrared sounder data and calculates atmospheric vertical temperature profiles for at least 32 layers															
2.1.1	The input to this code module shall be thinned radiances from the IASI sensor															
2.1.1.1	The IASI radiances shall be thinned as described in the <Project Name> ATBD.															
3.0	The reported temperature for each vertical layer shall have an accuracy of 0.5 K															
3.1	The code module that reads infrared sounder data and calculates atmospheric vertical temperature profiles for at least 32 layers shall achieve an accuracy of 0.5 K for each layer.															

Figure 7.1 – Example of a Requirements Vertical Traceability Matrix

Figure 7.1 is a simple example, containing 15 total requirements drivers and requirements. Even so, it cannot be easily read unless the reader zooms in to at least 200%. Typically, an actual project RVTM will be large, with as many rows and columns as the combined number of all requirements drivers and all requirements. For this reason, it is recommended that the RVTM be maintained as a Microsoft Excel spreadsheet that is included as an appendix to the RAD.

In the example of Figure 7.1, the binary condition of the cells is indicated by filled or blank cells. This choice is probably the most visually apparent. Note that the diagonal cells are patterned. This is a cosmetic choice to allow the reader to easily see the symmetric nature of the matrix. There is also a good reason for indicating the binary condition with zeroes or ones. When this is done, you can obtain a measure of the complexity of all requirements as the sum of all cells and also measures of the complexity of each requirement as the sum of its row or column. This can be useful for requirements management and risk assessment purposes. Note that you can store the binary condition of the cells in more than one way by using multiple sheets in the Excel file, In that way, you can have a sheet that displays the traceability visually for optimum accessibility and a separate sheet that displays the traceability numerically for optimum management and risk assessment.

Requirements Traceability includes traceability from a requirement to its allocation of functions, objects, people, processes, and work products. This is an extension of vertical traceability to the allocation, which can be thought of as a lower level extension of derived requirements. It is important to understand the distinction between derived requirements and requirements allocations, though. While requirements allocations are essentially design decisions that can be readily changed with no effect on the requirements themselves, derived requirements cannot be changed without assurance that basic requirements and requirements drivers are not at risk.

Traceability from a requirement to its allocations is established with the RAS (c.f. Section 6.4 of this TD). Some of this traceability may be established for PRR and documented in RAD v1r0, but it is expected that RAD v1r1 and v1r2 will provide substantial upgrades as the design matures for PDR and CDR.

7.1.2. Horizontal Traceability

Requirements Traceability includes horizontal traceability from function to function and across interfaces of the functional architecture. Most of this should be established for CDR and documented in RAD v1r2. Provide a Requirements Horizontal Traceability Matrix (RHTM) that summarizes the traceability from function to function and across interfaces.

It is recommended that horizontal traceability be documented in the RAD by a Requirements Horizontal Traceability Matrix (RHTM) that summarizes the two-way horizontal traceability between functions.

-

- **Document horizontal traceability in the RAD by a Requirements Horizontal Traceability Matrix (RHTM) that summarizes the two-way horizontal traceability between functions.**

-

The rows of the RHTM contain each function that has been identified by functional analysis and grouped by allocation of requirements to functions. It is recommended that functions be numbered for identification, in a fashion similar to the numbering of requirements (c.f.

Section 4.6 of this TD). This numbering should be accomplished during the Allocation to Functions sub-process of Requirements Allocation (c.f. Section 6.1 of this TD). If this is done, functions can be listed in the RHTM rows in numerical order, thereby establishing a logical structure to the RHTM that will enhance its ease of use.

The columns of the RHTM should be identical in content and order to the rows, thereby creating a square symmetric matrix whose diagonal elements are identities.

Each cell in the RHTM consists of a binary condition (e.g., yes or no, 0 or 1, filled or blank) that indicates whether the function of its column is traceable to the function of its row.

An example RHTM is shown as Figure 7.2.

		Function Number												
Function Number	**Function Statement**	**1.0**	**1.1**	**1.1.1**	**2.0**	**2.1**	**2.1.1**	**2.1.2**	**2.2**	**2.3**	**3.0**	**3.1**	**3.1.1**	**3.2**
1.0	Read IASI radiances from the operational Level 1 IASI data sets													
1.1	Produce code that reads the operational Level 1 IASI data formats													
1.1.1	There shall be a code module that reads infrared sounder data and calculates atmospheric vertical temperature profiles													
2.0	Extract the appropriate subset of the IASI data													
2.1	Thin the IASI radiances													
2.1.1	Spectral thinning of the IASI radiances													
2.1.2	Spatial thinning of the IASI radiances													
2.2	Apply a cloud mask													
2.3	Apply data quality masks													
3.0	Compute the atmospheric vertical temperature profile for 32 layers													
3.1	Apply the <RTM Name> RTM to compute model radiances for a standard set of vertical temperature profiles													
3.1.1	Fit the model radiances to observed radiances to derive best fit temperatures for each layer													
3.2	Write the derived atmospheric vertical temperature profiles to an output data set													

Figure 7.2 – Example of a Requirements Horizontal Traceability Matrix

7.2. Requirements Tracking

Requirements tracking involves the monitoring of the status of the requirements and their allocation to ensure that the integrity of the requirements allocation is preserved as the solutions, design and implementation matures through the Design and Build phases of the STAR EPL.

This is an activity that involves project management and project QA. Project management is involved as part of its normal activities of project monitoring and control (c.f. Section 9 of this TD). Project QA is involved as part of its normal activities of project QA. Project management should ensure that a system is in place for tracking requirements and requirements changes. Project QA should ensure that the system is followed during the project lifecycle.

RAD v1r1, v1r2 and v2r0 should include a report on the status of requirements tracking.

7.3. Requirements Validation

Requirements validation is concerned with ensuring that the requirements and requirements allocation (c.f. Section 6 of this TD) provide a satisfactory balance between customer/user needs and expectations, NESDIS mission goals, technical feasibility, the available resources and external constraints.

Requirements validation activities have already been addressed in Section 5 of this TD, in the context of Requirements Analysis. Because these activities are part of both Requirements Analysis and Requirements QA, we re-visit these activities in this subsection.

First, the validator of requirements should determine whether they are consistent with the NESDIS mission and strategic plan, as discussed in Section 5.1 of this TD. This analysis should begin with the review of the project proposal at Gate 2. A Gate 2 review is limited to analysis of the customer needs and product descriptions as stated in the proposal. As the project requirements are more fully developed, it is necessary to ensure that they are not diverging from the NESDIS mission and strategic plan. The project manager and Development Lead should ensure that project requirements validators either have access to this plan or are provided with guidelines derived from this plan.

Any identified conflicts between customer needs and expectations must be addressed and resolved before requirements development is completed. Ideally, this will occur at a Gate 2 Review. If necessary, this can be deferred to a Gate 3 Review. Because an operations concept may not be developed until the PRR, it is possible that conflicts will be discovered after the Gate 3 Review. In that case, it is a top priority that the requirements developers consult with project management and customers to resolve these conflicts as soon as possible. It is not acceptable for a project to go to its PRR with unresolved conflicts.

- **Identify and resolve conflicts between the NESDIS strategic and mission plan and the customer/user needs and expectations as soon as possible. Requirements acceptance analysis cannot be completed until all conflicts are resolved.**

- **Ensure that requirements are consistent with the NESDIS strategic and mission plan.**

Next, the validator of requirements should determine whether they are complete, feasible, realizable, and verifiable. This is part of the technical analysis of requirements, as discussed in Section 5.2 of this TD. While design determines the feasibility of a particular solution, technical requirements analysis addresses knowing which requirements affect feasibility. Identify key requirements that have a strong influence on cost, schedule, functionality, risk, or performance. Identify technical performance measures that will be tracked during the development effort.

Quantitative analysis, discussed in Section 5.3 of this TD, is a specialized subset of technical analysis that is focused on performance requirements. Performance requirements must be specific and quantitative.

- **Ensure that technical and quantitative analysis of requirements have determined whether they are complete, feasible, realizable, and verifiable.**

If all performance requirements are treated equally, an endless loop of tradeoffs addressing conflicting requirements and design may ensue. A priority based on customer desires and the NESDIS strategic and mission plan must therefore be established, as discussed in Section 5.5 of this TD. Requirements validators should ensure that requirements developers have identified key requirements that have a strong influence on cost, schedule, functionality, risk, or performance and have struck a balance between customer needs and expectations, whether quantitative or qualitative, and anticipated constraints. Consider cost, schedule and technical constraints. Consider the importance of the product performance to the NESDIS strategic and mission plan.

- **Ensure that requirements have been prioritized, based on customer desires, the NESDIS strategic and mission plan, and anticipated constraints.**

For PRR, validation of requirements should include a demonstration that a balance has been established between the basic requirements statements, customer/user needs and expectations, and constraints on the production, distribution and performance of products. This demonstration can be extended to derived requirements and requirements allocations that have been developed by PRR.

Derived requirements are validated by a demonstration that they are the best set of requirements to satisfy the basic requirements.

Requirements allocations are validated by a demonstration that the solution and design provides a feasible, satisfactory implementation for meeting the requirements.

It is desirable to accomplish requirements validation as early as possible in the product lifecycle, to prevent the need for re-design later if it becomes evident that requirements or requirements allocations must be significantly changed to redress a lack of balance between them, customer/user needs and expectations, and constraints on the production, distribution and performance of products.

At PDR, requirements allocation has typically been substantially upgraded by the preliminary design and derived requirements may be added or refined to respond to issues raised by the preliminary design. It is important to review the requirements validation at PDR to ensure that it is sufficient for the complete current set of requirements and requirements allocations.

At CDR, requirements allocation has typically been upgraded by the detailed design and derived requirements may be added or refined to respond to issues raised by the detailed design. It is important to review the requirements validation at CDR to ensure that it is sufficient for the complete current set of requirements and requirements allocations.

Requirements validation is performed early in the development effort to gain confidence that the requirements are capable of guiding a development that results in successful final validation. This activity should be integrated with risk management activities. Mature organizations will typically perform requirements validation in a more sophisticated way and will broaden the basis of the validation to include other stakeholder needs and expectations. These organizations will typically perform analyses, simulations, or prototypes to ensure that requirements will satisfy stakeholder needs and expectations.

Requirements validation plans and status should be addressed in the project's VVP. Typically, the stakeholders responsible for requirements validation are also responsible for contributing to the VVP. The requirements validation plan, as documented in the VVP, should follow STAR standards for requirements validation. Refer to DG-6.3 for VVP guidelines.

- **Document requirements validation plans and status in the project's Verification and Validation Plan (VVP).**

7.4. Requirements Verification

Requirements verification is concerned with ensuring that the requirements are identified, analyzed, validated and allocated in accordance with standard processes. In addition to the requirements being necessary, stated clearly and unambiguously, the requirements must be verifiable by a technique satisfactory to the customer. It is necessary to describe, at least at a preliminary level, the specific verification technique to be used for each requirement.

Verification methods include:

- **Inspection –** a verification method that uses a visual process to confirm that requirements are met. For example, a check to see if integration has installed all appropriate units.
- **Test –** a verification method that measures quantifiable dimensions (measured performance). Test may use electrical or mechanical instruments to help perform the measurements. For example, a test of the RF performance of a radar would entail measurement of the effective radiated power
- **Demonstration –** a verification method that uses the product in some operation that verifies the product's capability to meet operation objectives. Usually this is the "turn-on-the-TV" method to see if the product seems to perform as designed.
- **Analysis –** a verification method that uses models (through design, similarity, or analytical arguments) to perform verification. The models may be mathematical, narrative, graphical, or physical. For example, an analyst could determine the moment of inertia of a product through calculation rather than a specific measurement.

Verification of requirements development with respect to standards is usually by Inspection. The primary artifact to be inspected is the RAD.

As requirements are defined and refined through technical analysis, their relationship to higher level requirements and the higher level defined functionality (c.f. Section 5.4 of this TD) must be understood to ensure satisfactory requirements verification.

8. REQUIREMENTS MAINTENANCE

Requirements maintenance is an activity that is performed throughout all steps of the product lifecycle following requirements development. It is important that project stakeholders react to project variance by determining what effect it may have on the project requirements. Often, the requirements or their allocation must be changed to respond to project variance.

A project may adopt the STAR standard set of maintenance processes. In that case, it is sufficient to state this as the sole project maintenance requirement. If a project chooses to tailor its maintenance process, there must be derived system requirements that state how this process will be tailored.

After the requirements development process is essentially completed at the end of the Design phase of the STAR EPL, it is very important that new requests from customers or other requirements sources be scrutinized to avoid requirements creep. There may be valid reasons to accept new requirements at this stage of the product lifecycle, but these must be reviewed carefully before approval. This is a joint activity of requirements maintenance and requirements management (c.f. Section 9 of this TD). Technical and management review are usually both needed. If requested changes are sufficiently substantial, a new iteration of the requirements development process and a delta PRR may be necessary. Guidelines for dealing with such a circumstance are discussed in Section 9 of this TD.

9. REQUIREMENTS MANAGEMENT

Requirements management is an activity that is performed throughout all steps of the product lifecycle following requirements development. Requirements management activities are not exclusive to project managers, despite the terminology. Most project stakeholders will recognize activities outlined in this section that pertain to them.

Many of the activities outlined in this section have already been addressed in previous sections, because requirements management activities are imbedded in the other post-development requirements activities.

At a high level, requirements management activities encompass the following:

- Managing all changes to the requirements
- Maintaining the relationships between the requirements, the project plans, and the work products
- Identifying inconsistencies between the requirements, the project plans, and the work products
- Taking corrective action

To achieve this, the following requirements management practices should be adopted:

- Develop an understanding with the requirements providers on the meaning of the requirements.
- Obtain commitment to the requirements from the project participants.
- Manage changes to the requirements as they evolve during the project.
- Maintain bidirectional traceability among the requirements and the project plans and work products.
- Identify and correct inconsistencies between the project plans and work products and the requirements.

9.1. Understanding the Requirements

To avoid requirements creep, criteria should be established to designate appropriate channels, or official sources, from which to receive requirements (c.f. Section 4.1 of this

TD). Establish objective criteria for the acceptance of requirements. Examples of acceptance criteria include the following:

a. Clearly and properly stated
b. Complete
c. Consistent with each other
d. Uniquely identified
e. Appropriate to implement
f. Verifiable (testable)
g. Traceable

- **Designate appropriate channels, or official sources, from which to receive requirements**

- **Establish objective criteria for the acceptance of requirements.**

Conduct analyses of the requirements with the requirements provider to ensure that a compatible, shared understanding is reached on the meaning of the requirements. The result of this analysis and dialog is an agreed-to set of requirements that project stakeholders can commit to.

- **Conduct analyses of the requirements with the requirements provider to ensure that a compatible, shared understanding is reached on the meaning of the requirements.**

9.2. Committing to the Requirements

Obtain commitment to the requirements from the project stakeholders. It is essential that all project stakeholders who are affected by specific requirements commit themselves to the activities that will be needed to satisfy them. These activities should be documented in a project plan.

- **Obtain commitment to the requirements from the project stakeholders.**

Commitment to the requirements should be established by the PRR, but the extent of the commitment does not end there. Typically, requirements will change during the design development steps of the process, driven either by constraints discovered during design development or by new requirements requested by customers and users. These changes will have an impact on project plans, activities and work products. The STAR EPL includes practices designed to facilitate enduring stakeholder commitment to the requirements, by including project stakeholders in every technical review and requiring acceptance of changes to requirements and requirements allocation at every review. It is the responsibility of requirements management to obtain the commitment of project stakeholders to these practices and then to ensure that these practices are implemented.

9.3. Changing the Requirements

As noted in the previous subsection, requirements and their allocation to system and product components will typically change during the development lifecycle. Changes are expected during design development and may also be necessary during code development and testing.

Management of these changes involves the following:

- Capture all requirements and requirements changes that are given to or generated by the project. It is necessary that the source of each requirement is known and the rationale for any change is documented.
- Maintain the requirements change history with the rationale for the changes. It is necessary that the rationale for any change is documented. The project manager may want to track appropriate measures of requirements volatility to judge whether

new or revised controls are necessary. Maintaining the change history helps track requirements volatility.

- Evaluate the impact of requirement changes from the standpoint of relevant stakeholders.
- Make the requirements and change data available to the project.

-

- **Manage changes to project requirements and requirements allocations as they evolve throughout the project.**

9.4. Tracing the Requirements

When the requirements are managed well, traceability can be established from the source requirement to its lower level requirements and from the lower level requirements back to their source (c.f. Section 7.1 of this TD). Requirements traceability can also cover the relationships to other entities such as intermediate and final work products, changes in design documentation, test plans, and work tasks.

-

- **Maintain bidirectional traceability among the requirements and the project plans and work products.**

The traceability should cover both the horizontal and vertical relationships, such as across interfaces. Generate and maintain traceability matrices.

Traceability is particularly needed in conducting the impact assessment of requirements changes on the project plans, activities, and work products.

9.5. Tracking the Requirements

Maintain a system for tracking requirements and requirements changes.

- **Maintain a system for tracking requirements and requirements changes.**

Requirements tracking (c.f. Section 7.2 of this TD) involves the monitoring of the status of the requirements and their allocation to ensure that the integrity of the requirements allocation is preserved as the solutions, design and implementation matures through the Design Development and Code Development phases of the STAR EPL.

This is an activity that involves project management and project QA. Project management should ensure that a system is in place for tracking requirements and requirements changes. Project QA should ensure that the system is followed during the project lifecycle.

Review the project's plans, activities, and work products for consistency with the requirements and the changes made to them. Find the inconsistencies between the requirements and the project plans and work products. Identify changes that need to be made to the plans and work products resulting from changes to the requirements baseline. Implement the necessary changes. Verify that the necessary changes have been satisfactorily made.

- **Review the project's plans, activities, and work products for consistency with the requirements and the changes made to them.**

- **Find the inconsistencies between the requirements and the project plans and work products.**

-

- **Identify changes that need to be made to the plans and work products resulting from changes to the requirements baseline. Implement the necessary changes. Implement the necessary changes. Verify that the necessary changes have been satisfactorily made.**

Note potential effects of the requirement changes on the project plan. Identify and evaluate risks associated with each requirement change. The risks should be reviewed and possible actions generated at PRR and subsequent reviews. Ensure that project plans are updated to be consistent with all approved requirements changes.

-

- **Note potential effects of the requirement changes on the project plan.**

-

- **Identify and evaluate risks associated with each requirement change.**

Each revision of the RAD should update the status of risks that were documented in the previous RAD version.

APPENDIX A – QUALITY FUNCTION DEPLOYMENT

This Appendix provides a description of the Quality Function Deployment (QFD) method for capturing, prioritizing, and translating customer requirements into team selected project requirements. This description is adapted from a Raytheon Systems Engineering document (5-SE-H0004) and tailored for the STAR organization.

The QFD process uses a series of matrices to accomplish a requirements translation. Beginning with the House of Quality, customer requirements are systematically cascaded through design definition, manufacturing evaluation and process development, and production improvements. This is accomplished by creating a cascaded chart in which the HOWs of the previous chart become the WHATs of the next level chart. Those HOWs that are important, or high risk are typically taken to the next level of detail.

This cascading process continues until customer objective is refined to an actionable level. Specific project requirements will determine the number of matrices to be completed. The matrices are requirements documentation by which the customer requirements are driven throughout the project. It is the analysis of the data found within these matrices that is important.

A model of the House of Quality is found in Figure A.1. This model represents the basic structure of all matrices used within the QFD process. Sections 1, 2, 4, 5, 7, 9, and 10 are considered essential to have an effective QFD performed. Section 3 is not relevant to the STAR organization. Figure A.1 will serve as a reference to the procedural steps that follow.

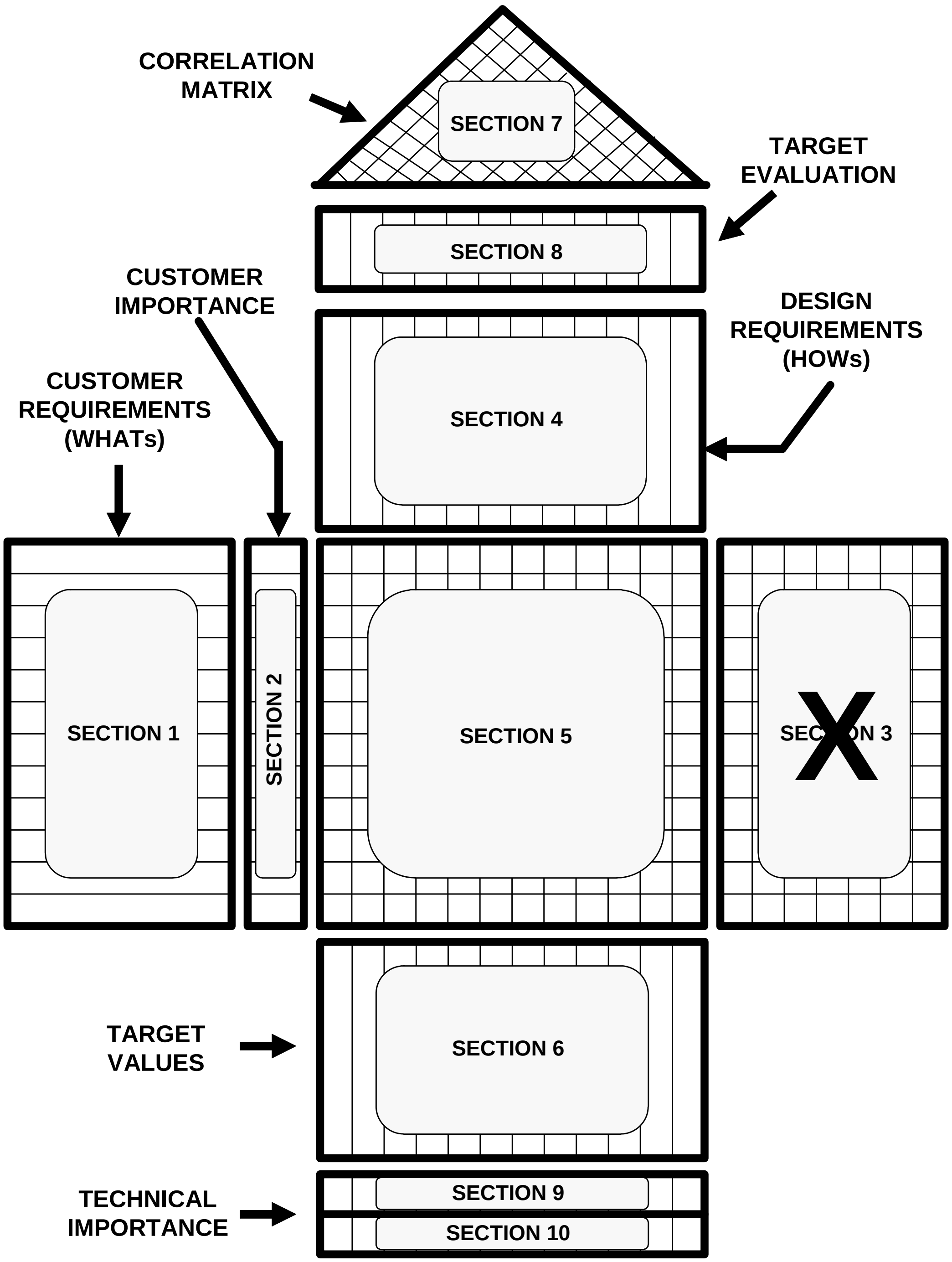

Figure A.1 – QFD House of Quality Matrix

The QFD standard method consists of ten steps:

1. ***Identify Customer Requirements (WHATs)*** – QFD does not specify how to identify the WHATs. In the STAR EPL process, this is accomplished through the development of an operations concept and basic requirements driven by customer needs and expectations.

2. ***Determine Customer Importance*** – Customer importance values document how important the requirements are to the customer. The importance values are expressed as a relative scale (typically a spread of 1-5 or 1-10) with the higher numbers indicating greater importance to the customer. It is important to use the whole scale spread when performing this activity. Determine these customer's importance values and place them in section 2.

3. ***Establish the HOWs*** - As HOWs are developed, they should meet several key conditions. First, they should be measurable. Second, the requirements should not impose a particular design approach. Third, they should be developed with knowledge and understanding of the WHAT requirements. Identify the organization's HOWs and place them in section 4.

4. ***Complete the Relationship Matrix of WHATs vs. HOWs*** - The relationship matrix identifies the existence and degree of relationship between WHATs and HOWs. Specifically, this matrix addresses direct relationships in which the HOWs satisfy the WHATs in a strong, moderate, or weak manner. A standard numbering or symbology system (See Figure A.2) is used to indicate the degree of this relationship. If there is no direct relationship, the cell is left blank. Data indicates that completing the relationship matrix in a column-wise fashion works best. Evaluate each of the HOWs against each of the WHATs and indicate the degree of relationship that exists between them at the point of intersection in section 5.

5. ***Establish Targets for the HOWs*** - Targets are those values for the HOWs that are expected to achieve customer satisfaction. As such, these targets may differ from design specifications. Identify target values for each of the HOWs and place them in section 6. Identify the orientation of each of the HOWs and place them in section 8. Orientation refers to the "ideal direction to optimize each of the HOWs." By identifying the orientations, a better understanding of the HOWs is achieved. Orientations are identified in three ways:

(a) Maximize
(b) Minimize
(c) Target

6. ***Complete the Correlation Matrix*** - The correlation matrix is the roof-like structure located at the top of the QFD matrix. Once filled in, the correlation matrix indicates where trade-off decisions may be required. To complete this matrix, each design requirement is examined against every other design requirement. Where correlations exist between two different requirements, place either a symbol or number at the point of intersection of those two requirements in section 7. The symbols or numbers used in the correlation matrix (See Figure A.2) should represent the degree of relationship that exists between the two requirements, i.e., strong positive, positive, negative, or strong negative. If there is no direct correlation, the cell is left blank.

7. ***Calculate the Technical Importance*** - The technical importance values combine the degree of importance to the customer with the strength of the relationship that exists between the HOWs and the WHATs. The first of these values is calculated by multiplying the customer importance factor in section 2 by each relationship value in section 5 and then summing these products for each column of HOWs. Each final product is placed in section 9 of the matrix in the row entitled "Absolute." The second value is a percentage allocation that is calculated by totaling all the values in the absolute row and then determining their relative proportions by dividing each absolute value by the total. This row may not always add to 100 percent due to mathematical rounding. This field is section 10 on the matrix.

8. ***Risk Assessment*** - The key benefit of using QFD is the connection of customer requirements (WHATs) to the project's options, product functions or product features (HOWs). However, the concise matrix structure of QFD supports the introduction of other metrics for better overall planning. For example, if the project is using QFD for concept development, it is very useful to introduce a measure of risk assessment for each HOW used in the matrix. As the technical importance of each HOW is developed, the measure of risk assessment could guide deployment strategy or at least identify those important measures where the project must deploy an effective risk mitigation plan. This measure of risk can be numeric, symbolic, or graphic. This number may be obtained as a result of formal risk methods. It is best to place it in a row close to the technical importance of the HOWs or the rank number of the HOWs. This will place the risk measure in relation to the importance of the HOW.

9. ***Analyze the Completed Chart*** - Analysis is the final step in completing the QFD matrix. The results of this analysis step will identify specific areas of importance that require additional attention. The following items represent a sample of typical things to look for when performing an analysis of any QFD matrix:

9.1 Blank rows that indicate unfulfilled customer wants.

9.2 Blank columns that indicate unnecessary requirements.

9.3 Rows or columns with only weak relationships.

9.4 HOWs that are not measurable. It's difficult to do what can't be measured.

9.5 Opportunities to exceed customer expectations.

9.6 Negative correlations that can be eliminated or improved.

9.7 Important, difficult, or high risk HOWs that will be deployed to the next level of detail.

9.8 Areas of customer importance where the competitor is not meeting the demand well enough.

9.9 Significant correlations between HOWs.

10. ***Completion of Subsequent QFD Charts*** - The House of Quality contains the most important information regarding the relationship between the product and the customer expectations. However, the completion of additional matrices may be required to ensure that these expectations are “heard” throughout the organization. Figure 1 provides a useful construct for additional matrices. Further guidance for the use of other matrix combinations may be gathered from the references. All subsequent matrices in the QFD process, are completed in the same fashion as outlined in procedural steps 1 - 9 above. A completed House of Quality matrix is seen in Figure A.2. This is a simple example for illustrative purposes only.

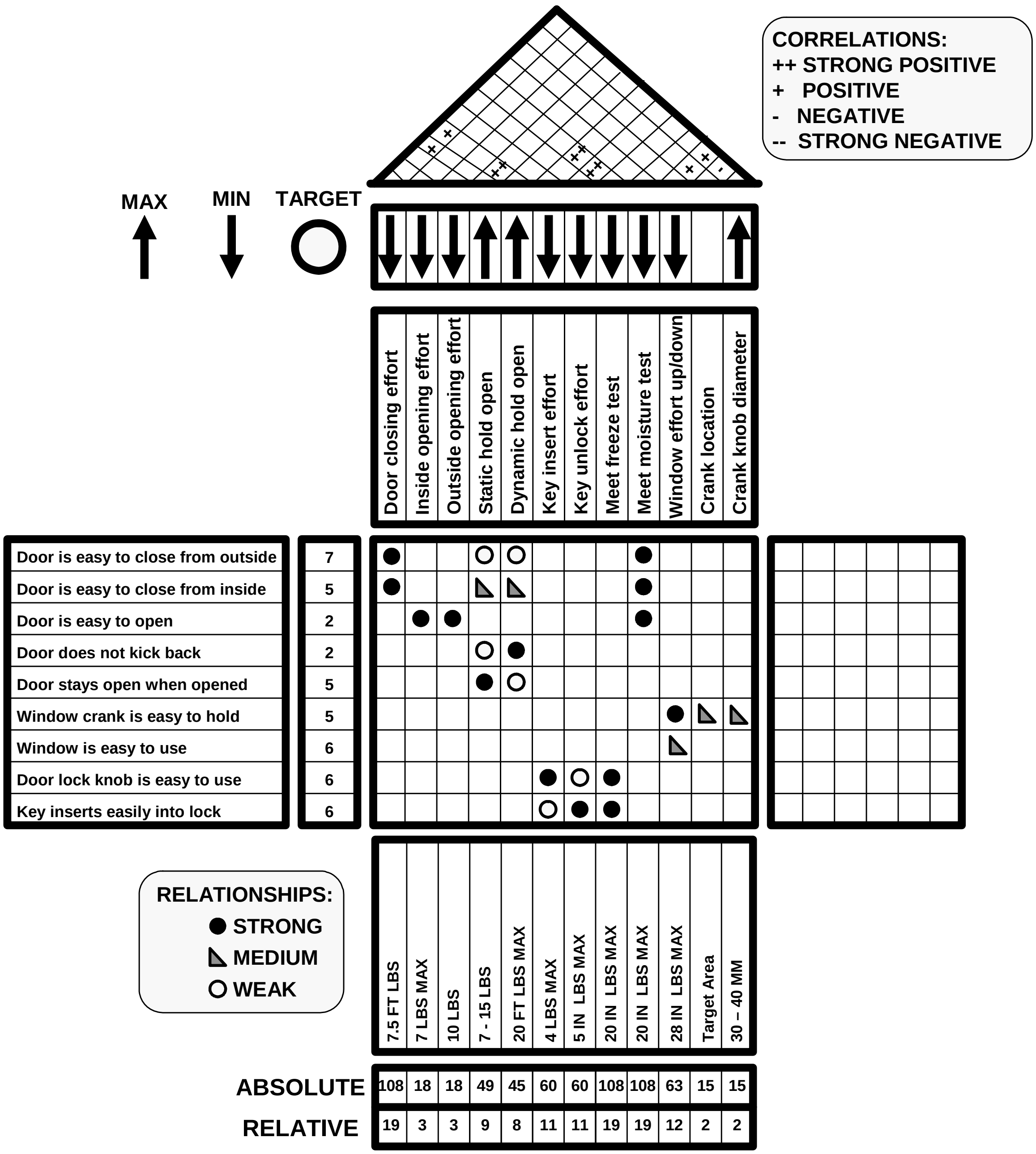

Figure A.2 – Completed QFD House of Quality Matrix

END OF DOCUMENT