# Fundamentals of Systems Engineering

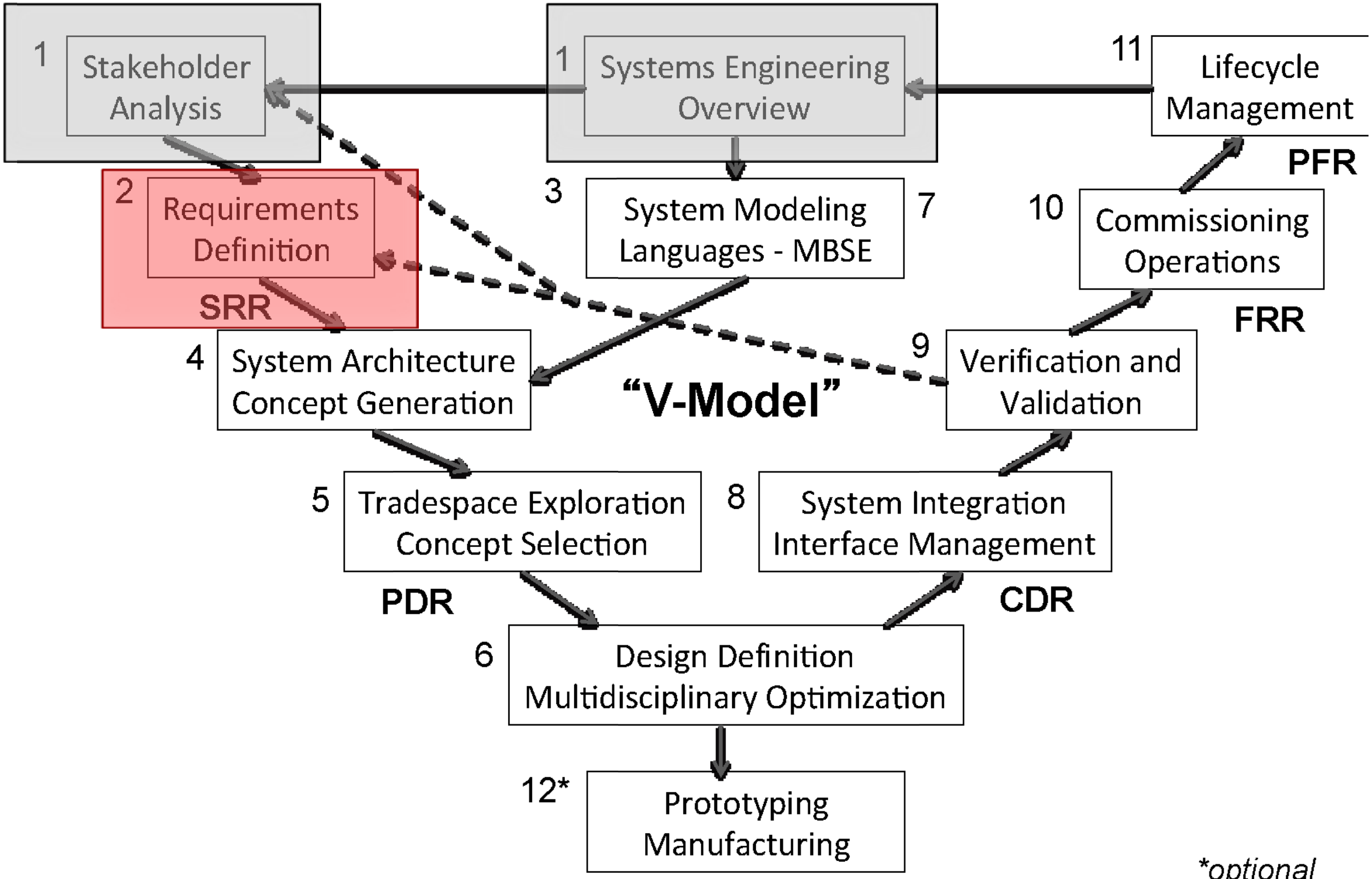
Prof. Olivier L. de Weck

## Session 2
Requirements Definition

*How we should specify "exactly" what is expected before we start designing something*

# The "V-Model" of Systems Engineering

16.842/ENG-421 Fundamentals of Systems Engineering




Numbers indicate the session # in this class

# Overview

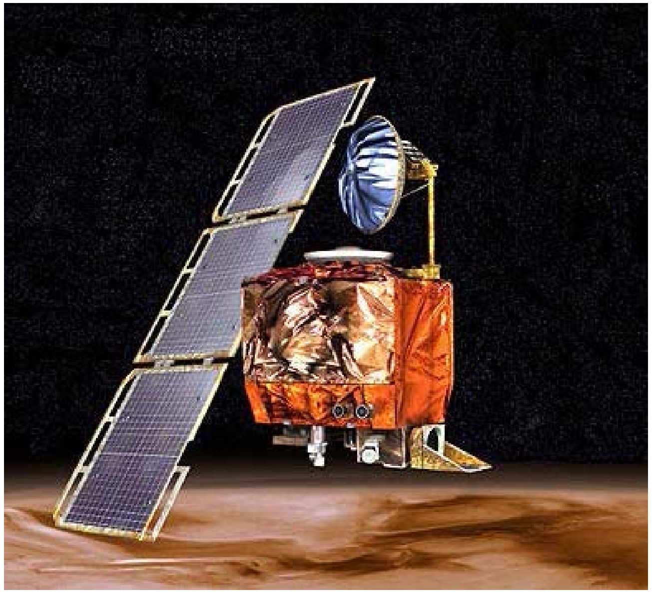
- What are requirements?
  - Definition, Examples, Evolution, Standards
- NASA Requirements Process
- Challenges of Requirements Definition
  - Flowdown and Allocation → Isoperformance
  - Validation and Verification
  - Writing good requirements
- What happens at the SRR?
- Kickoff Assignment A2

# Requirements Definition

- Requirements describe the **necessary functions and features of the system** we are to conceive, design, implement and operate.
  - Performance
  - Schedule
  - Cost
  - Other Characteristics (e.g. lifecycle properties)
- Requirements are often organized hierarchically
  - At a high level requirements focus on <u>what</u> should be achieved, not how to achieve it
  - Requirements are specified at every level, from the overall system to each hardware and software component.
- **Critically important to establish properly**
  - **Many of the cost overruns presented in Lecture 1 are caused by over-ambitious or missing requirements**

# Poor requirements example: MCO

- Mars Climate Orbiter (MCO) was launched by NASA on December 11, 1998
- Intended to study Martian climate, weather and surface changes and act as communications relay back to Earth
- **However, disintegrated during orbit insertion on Sept 23, 1999 → approach too close → requirements not followed**
- Units confusion problem: Ground Software produced output in non-SI units (lbf-sec) instead of SI units: Ns
  - Calculation of total momentum produced by engine burns needed by GNC
- Contract between NASA and Lockheed Martin did specify SI-units
  - This requirement was flowed down to the Software Interface Specification (SIS), but not verified later and not implemented in the AMD



This image is in the public domain.

From the accident report:

*"Items that the mission assurance manager could have addressed for MCO included ensuring that the AMD file met the requirements of the SIS .."*

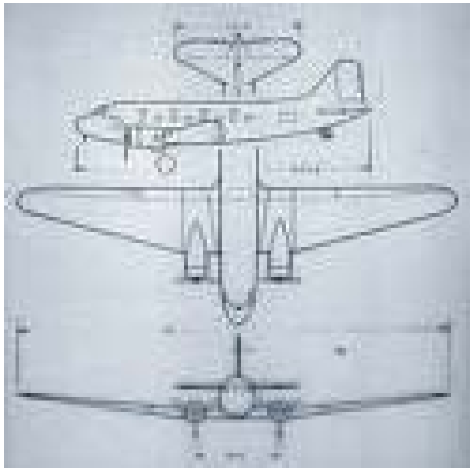
ftp://ftp.hq.nasa.gov/pub/pao/reports/1999/MCO_report.pdf

# Good requirements example: DC-3

- Requirements based on desired improvements to DC-2
- Very simple
  - 3 page RfP (McDonnell Museum)
  - Marathon phone call between Smith and Douglas
- Key Requirements
  - Range: 1000 miles
  - Cruise Speed: 150 mph
  - Passengers: 20-30
    - Depending on configuration
  - Twin Engines
  - Rugged and Economical

1st flight: 17 Dec 1935
Over 10,000 built

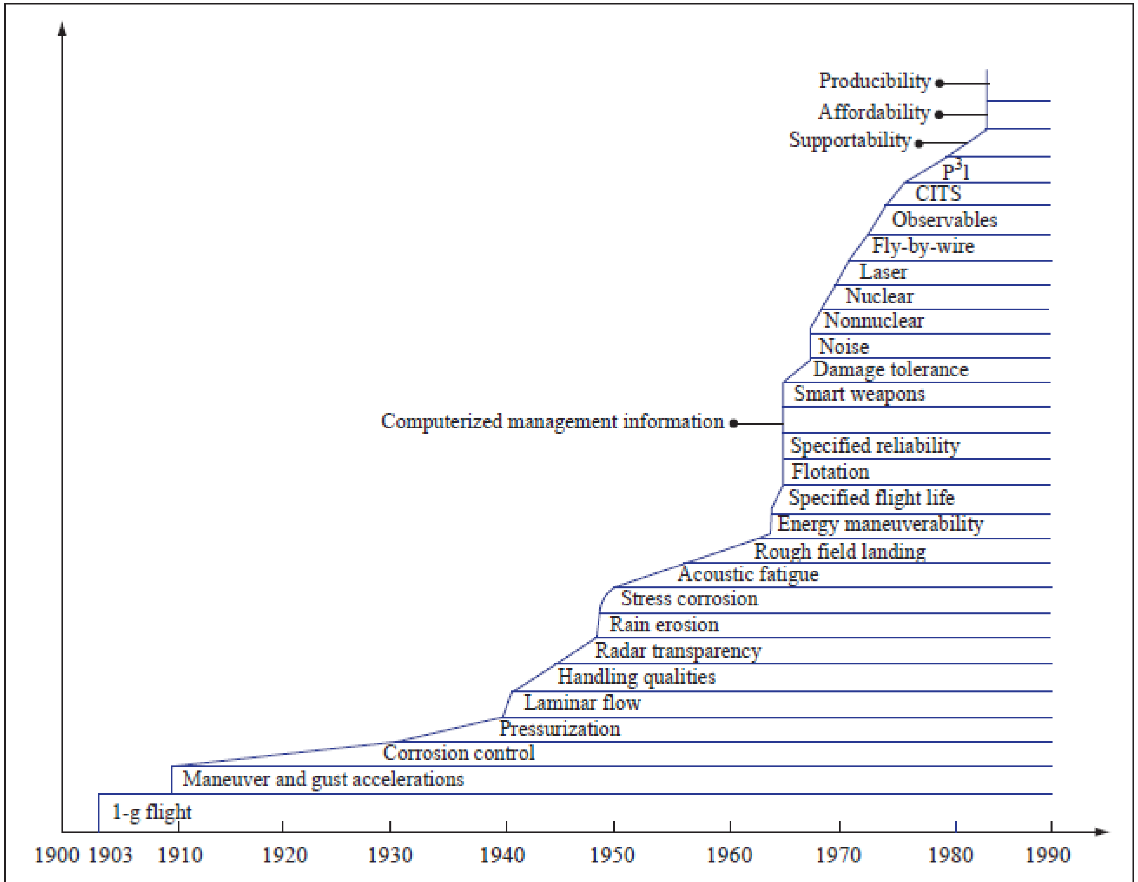
© source unknown. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.



© source unknown. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

# Requirements Explosion since 1970s

More and more requirements were added as systems grew in performance and complexity




Design requirements growth for aerospace vehicles.

Image by MIT OpenCourseWare.

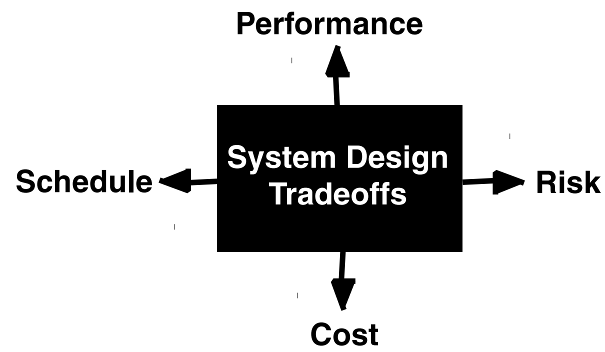
Source:
AIAA MDO TC
White Paper, 1991

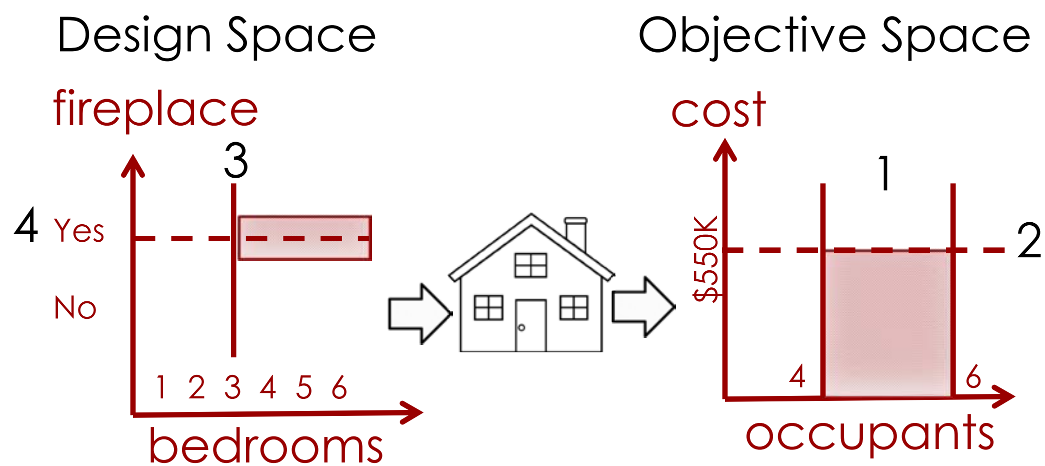
# Requirements Standards

- NASA Systems Engineering Handbook
  - NASA/SP-2007-6105
    - Section 4.2 (pp. 40-48) – Technical Requirements Definition
    - Section 6.2 (pp. 131-135) – Requirements Management
    - Appendix C (pp. 279-281) – How to write a good Requirement
    - Appendix D (pp. 282-283) – Requirements Verification Matrix
- International Council of Systems Engineering (INCOSE)
  - Systems Engineering Handbook, Version 3.1
  - Requirements Working Group
    - http://www.incose.org/ChaptersGroups/WorkingGroups/processes
- ISO/IEC 15288 (IEEE STD 15288-2008)
  - Systems and software engineering —
  - System life cycle processes
    - 6.4.1 Stakeholder Requirements Definition Process

# Requirements set *constraints* and *goals* in the design and objective space

- When designing systems we always have tradeoffs between performance, cost, schedule and risk
- "**Shall**" … Requirements help set constraints and define the boundaries of the design space and objective space
- "**Should**" … requirements set goals once "shall" requirements are satisfied
- Two main spaces:
  - Design *Space – the things we decide as engineers*
  - *Objective Space* – the things our systems/products achieve and what our customers care about




1. "The house shall sleep between 4 and 6 people"
2. "The total build cost should be less than $550K"
3. "The house shall have at least 3 bedrooms"
4. "The house should have a fireplace"

# Concept Question 3

- Is there any difference in meaning between the words "*Requirements*" and "*Specifications*"?
  - No, they are essentially the same.
  - Yes, requirements are the input to the design process, while specifications are the output.
  - Yes, specifications include the requirements, but also contain other things such as blueprints etc…
  - I am not sure.

- Answer Concept Question 3 (see supplemental files)

# Requirements vs. Specifications

- Requirements specify what the product or system shall/should do:
  - **Functions** it shall perform
  - How well it should perform these
  - Degree of automation of the system (what operators must do)
  - Compatibility with other devices etc…

- Specifications describe how the system is built and works
  - The **Form** it is made of
    - Materials used in the system
    - Overall dimensions
  - Schematics, Blueprints etc…
  - User Interface

"Description"

- Large enough to accommodate big dishes
- 1,200 Watts of power to reheat food quickly
- One touch settings for different food types (rice, pizza, frozen meals) …
- Etc…

Kenmore Elite Countertop 2.2 cu ft Microwave Oven

"Specification"

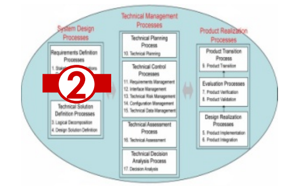
- Stainless steel exterior
- Dimensions: 24" x 14" x 19"
- Weight: 45.5 lbs
- General warranty: 1 year
- Power cord: included
- Etc …

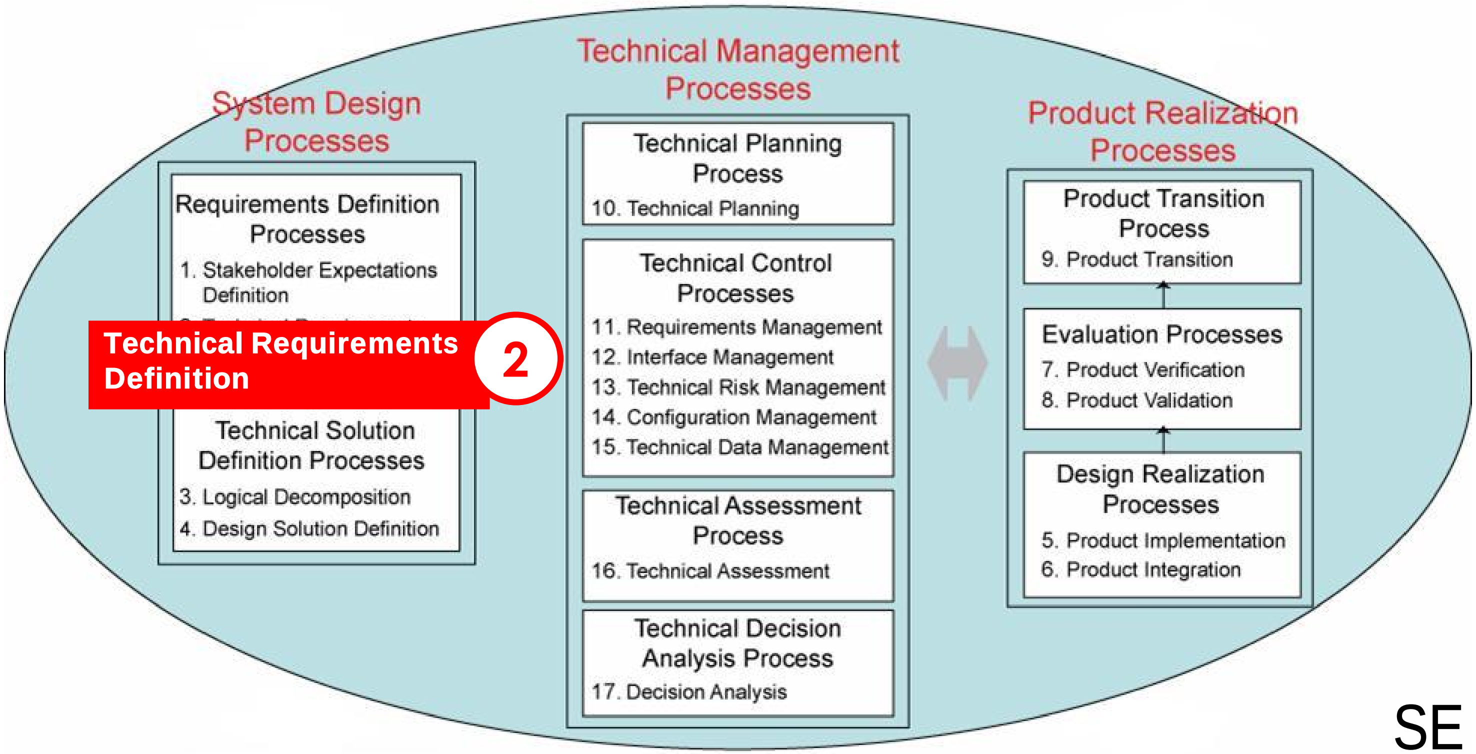
© source unknown. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

# Overview

- What are requirements?
  - Definition, Examples, Evolution, Standards
- NASA Requirements Process
- Challenges of Requirements Definition
  - Flowdown and Allocation → Isoperformance
  - Validation and Verification
  - Writing good requirements
- What happens at the SRR?
- Kickoff Assignment A2

# Technical Requirements Definition Process

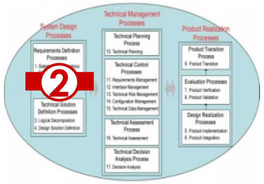




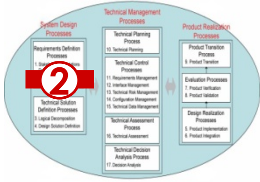

SE Engine

- **Requirement 16 (Section 3.2.2.1) "The Center Directors or designees shall establish and maintain a process, to include activities, requirements, guidelines, and documentation, for definition of the technical requirements from the set of agreed upon stakeholder expectations for the applicable WBS model."**

# Technical Requirements



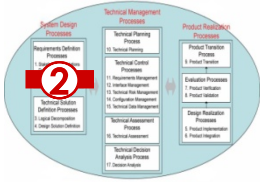
A cartoon removed due to copyright restrictions.



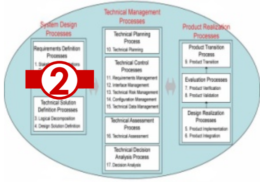
# Purpose of Technical Requirements Definition

- The Technical Requirements Definition Process
  - Is used to transform the baselined stakeholder expectations (input) into unique, quantitative, and measurable technical requirements (output)
- Requirements
  - Come in many flavors
  - Should be expressed as well-written “shall” statements that can be used for defining a design solution

# Importance of Technical Requirements Development (1/2)



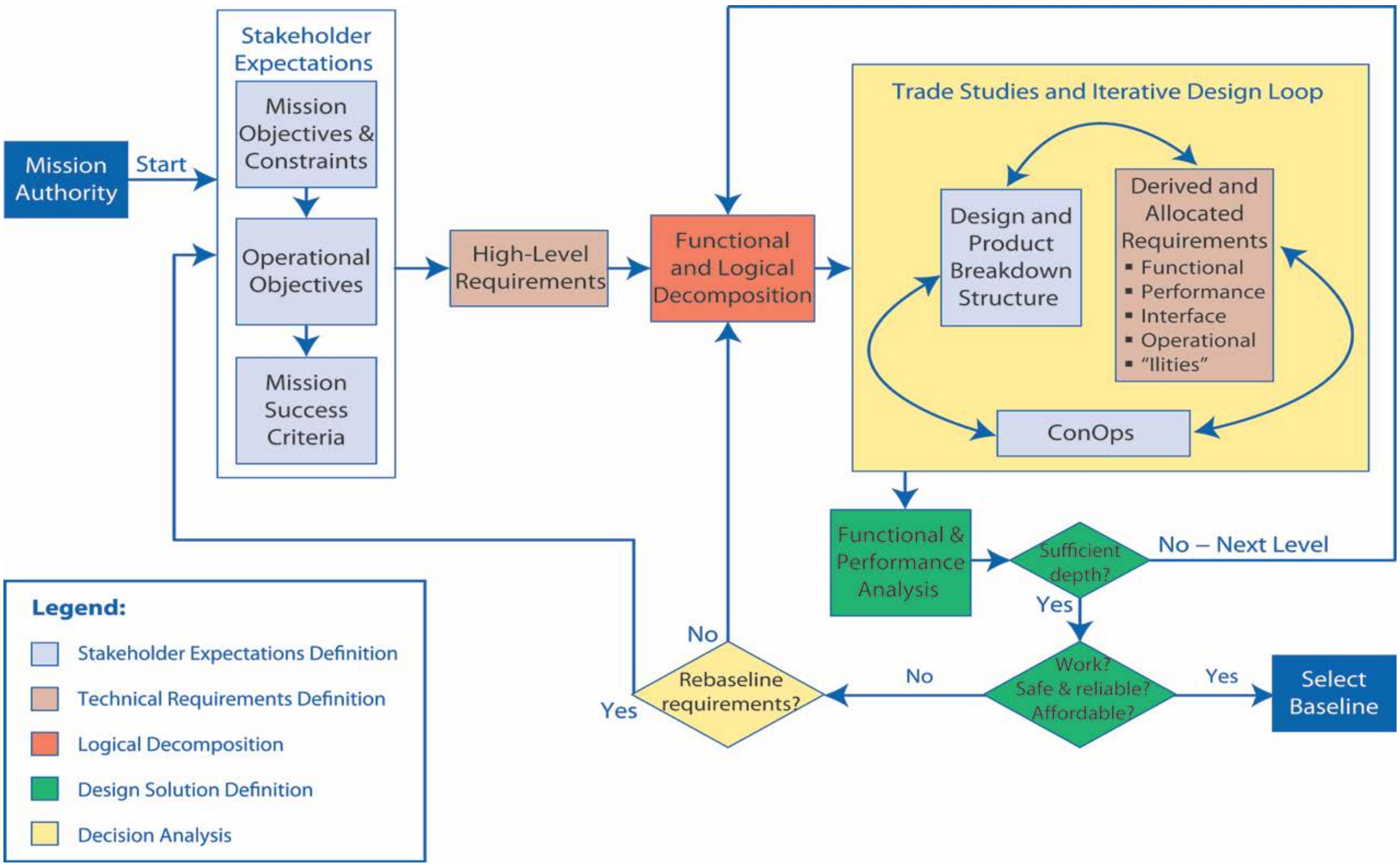
- **Establishes the basis for agreement between the stakeholders and the developers on what the product is to do**
- **Reduces the development effort because less rework is required to address poorly written, missing, and misunderstood requirements.**
  - Forces the relevant stakeholders to consider rigorously all of the requirements before design begins
  - Careful review can reveal omissions, misunderstandings, and inconsistencies early in the development cycle
- **Provides a basis for estimating costs and schedules**
  - The description of the product to be developed as given in the requirements is a realistic basis for estimating project costs and can be used to evaluate bids or price estimates



# Importance of Technical Requirements Development (2/2)

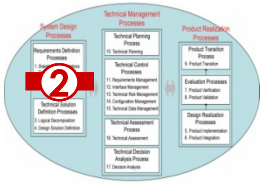
- **Provides a baseline for verification**
  - Organizations can develop their validation and verification plans much more productively from a good requirements document.
  - The requirements document provides a baseline against which compliance can be measured.
  - The requirements are also used to provide the stakeholders with a basis for acceptance of the system.
- **Facilitates transfer of the product to new users.**
- **Serve as a basis for later enhancement or alteration of the finished product.**

# Interrelationships Among the System Design Processes



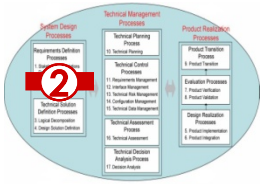

**SP-2007-6105, Figure 4.01**

This graph is in the public domain.




# Types of Requirements

- **Functional Requirements define what functions need to be done to accomplish the mission objectives**
  - Example: The Thrust Vector Controller (TVC) shall provide vehicle control about the pitch and yaw axes.
    - This statement describes a high level function that the TVC must perform.
    - Statement has form of Actor – Action Verb – object acted on
- **Performance Requirements define how well the system needs to perform the functions**
  - Example: The TVC shall gimbal the engine a maximum of 9 degrees, +/- 0.1 degree
- **Constraints are requirements that cannot be traded off with respect to cost, schedule or performance**
  - Example: The TVC shall weigh less than 120 lbs.
- **Interface Requirements**
  - Example: The TVC shall interface with the J-2X per conditions specified in the CxP 72262 Ares I US J-2X Interface Control Document, Section 3.4.3.
- **Environmental requirements**
  - Example: The TVC shall use the vibroacoustic and shock [loads] defined in CxP 72169, Ares 1 Systems Vibroacoustic and Shock Environments Data Book in all design, analysis and testing activities.
- **Other -illities requirement types described in the SE Handbook include: human factors, reliability requirements, and safety requirements.**



# Attributes of Acceptable Requirements

- A complete sentence with a single “shall” per numbered statement
- Characteristics for each Requirement Statement:
  - Clear and consistent – readily understandable
  - Correct – does not contain error of fact
  - Feasible – can be satisfied within natural physical laws, state of the art technologies, and other project constraints
  - Flexibility – Not stated as to how it is to be satisfied
  - Without ambiguity – only one interpretation makes sense
  - Singular – One actor-verb-object requirement
  - Verify – can be proved at the level of the architecture applicable
- Characteristics for pairs and sets of Requirement Statements:
  - Absence of redundancy – each requirement specified only once
  - Consistency – terms used are consistent
  - Completeness – usable to form a set of “design-to” requirements
  - Absence of conflicts – not in conflict with other requirements or itself

# Requirements Writing "Workshop"

- **Turn to your partner exercise (5 min)**
- Together write a good requirement that was (possibly) used in the development of one of the following solutions
  - A – Mr. Sticky tape for trapping flies
  - B – New BMW i3 electric car
  - C – EPFL Rolex Center

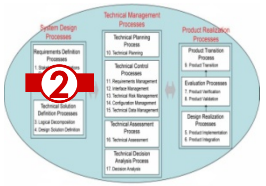

A

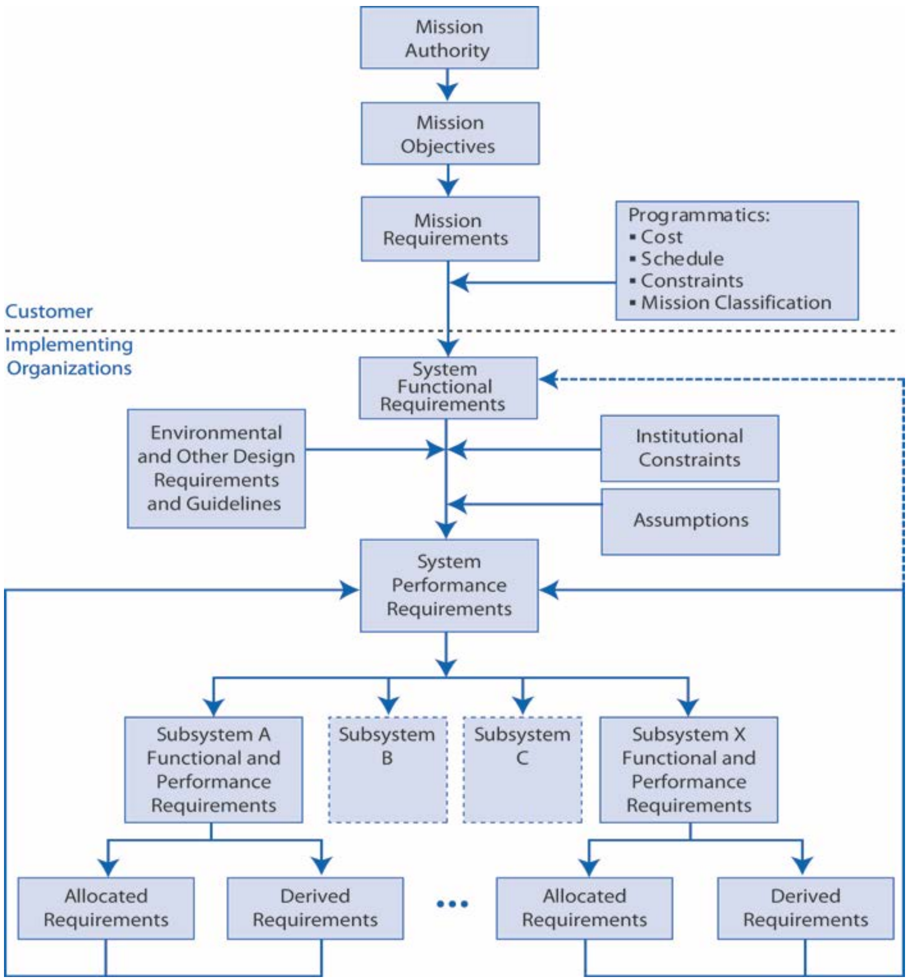
B

C

© sources unknown. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

# Requirements Decomposition, Allocation and Validation






**Source: SE HB Figure 4.2-3**

This graph is in the public domain.

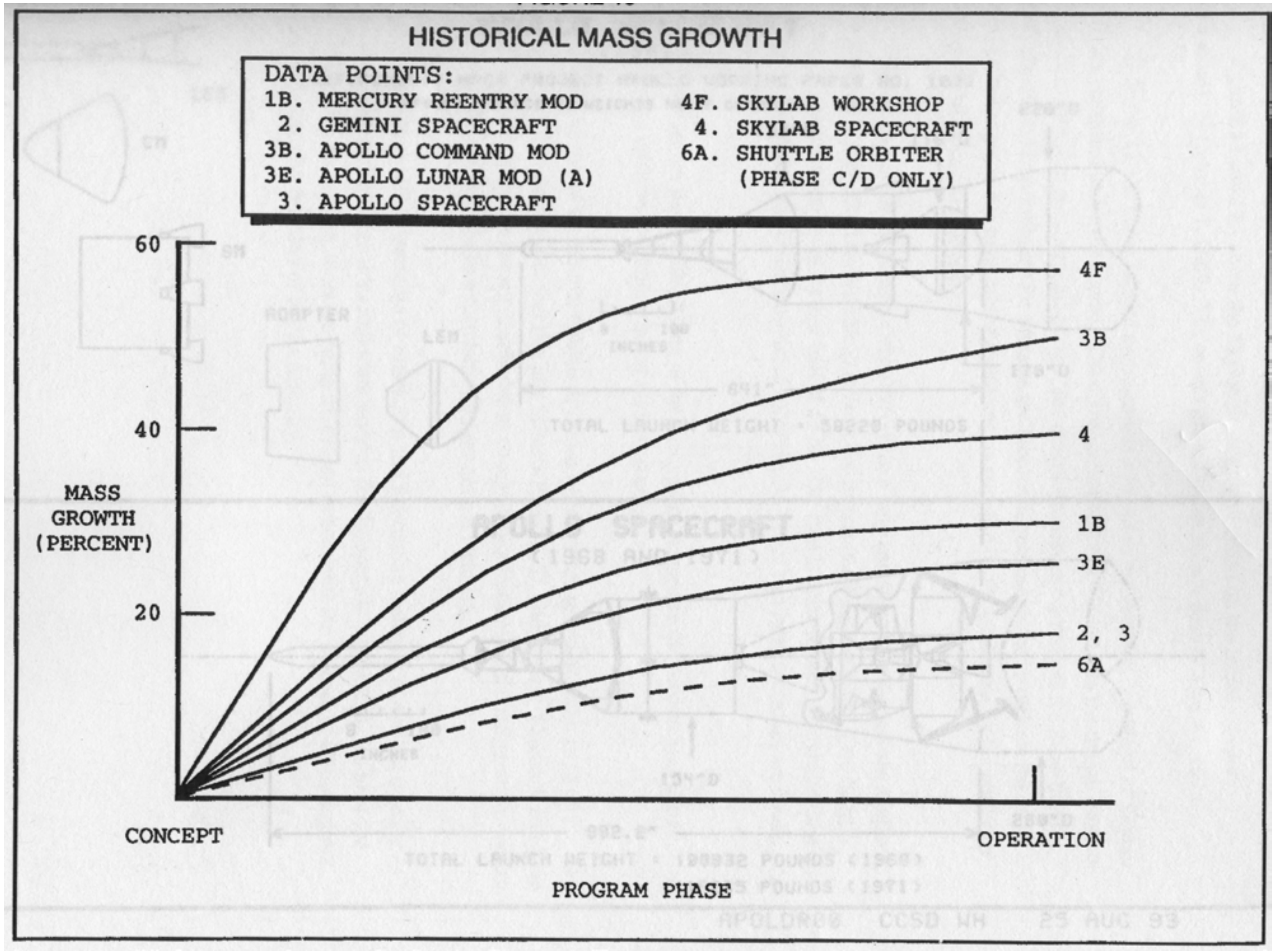
- Requirements are decomposed in a hierarchical structure starting with the highest level requirements.
- These high-level requirements are decomposed into functional and performance requirements and allocated across the system.
- These are then further decomposed and allocated among the elements and subsystems. This complete set of design-to requirements is achieved.
- At each level of decomposition (system, subsystem, component, etc.), the total set of derived requirements must be validated against the stakeholder expectations or higher level parent requirements.

# Requirements Margins Management

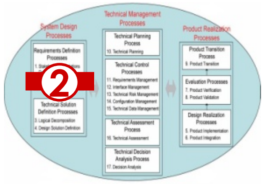
- Due to uncertainty during early design, must use appropriate requirements margins (e.g. for mass, power, memory etc…)
- Margins are *reserves*, that are not allocated to subsystems but are controlled by project managers
- For example mass growth has been experienced in almost all vehicle development programs
  - Mass growth can typically range from 10-60%
  - Depends on *novelty* of the project
- Typical Guidelines:
  - Establish SRR Mass + 30% margin
  - pre-PDR keep 20% mass margin
  - pre-CDR keep 10% mass margin
  - pre-IOC keep 5% mass margin

# Historical Mass Growth in Manned Spacecraft

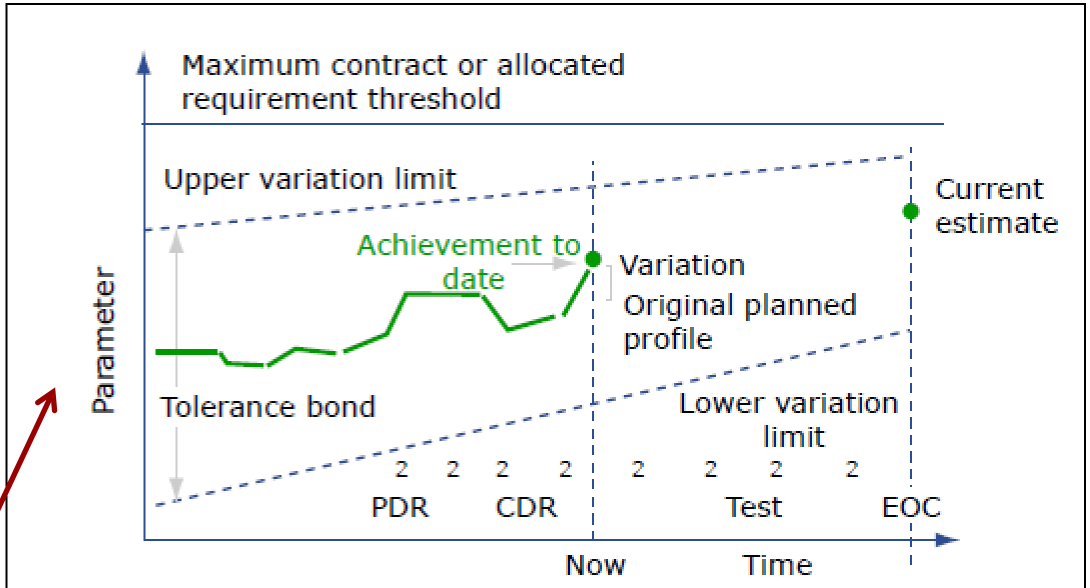
- Source: NASA JSC-26098 (Nov 1994) for Manned Spacecraft (does not yet include CEV Orion )




This graph is in the public domain.



# Monitoring a Requirement

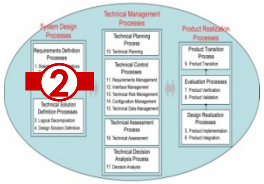


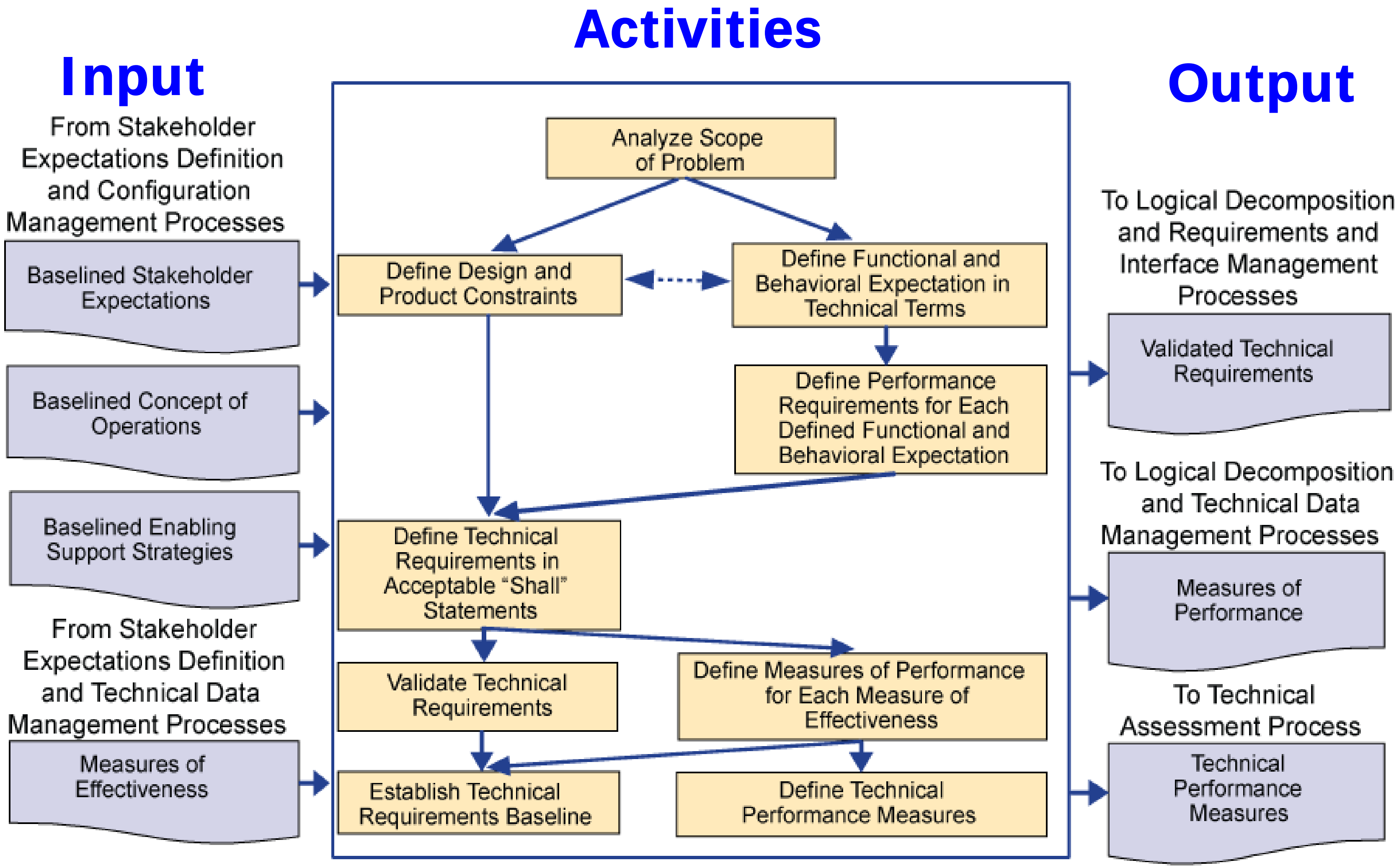

Technical performance measure (TPM)

| Development phase | Measurement |
|---|---|
| *Through PDR* | *Estimated value* |
| *PDR - CDR* | *Allocated value* |
| *CDR - Test* | *Calculated value* |
| *Test - delivery* | *Measured value* |

Image by MIT OpenCourseWare.

# Technical Requirements Definition Best Practice Process Flow Diagram

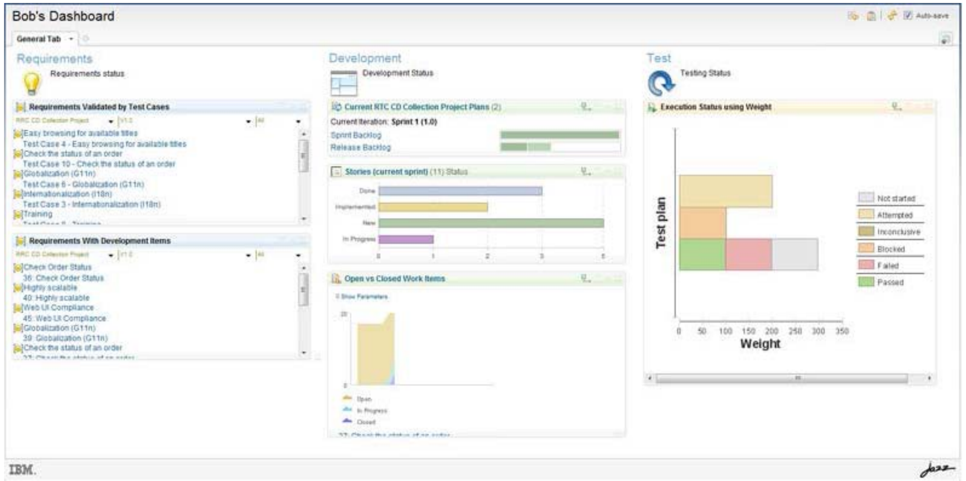





This image is in the public domain.

# Requirements Capture: Documents vs. Database

- Where / how are requirements captured?
  - Low cost “easy” solution: Create a **document** (e.g. in MS Word or Excel) to capture and revise the requirements. Use hyperlinks to link requirements.
  - This works okay for smaller projects with dozens or a few hundred requirements (e.g. about 3 levels of decomposition→ $(7+/-2)^3 = 125 - 729$
  - For larger projects with >1,000 requirements need to use a relational database
  - **Commercial Tools**, e.g. DOORS are available (but can be expensive)




© IBM. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

© IBM. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

IBM Rational DOORS
http://www.ibm.com/developerworks/rational/library/rational-doors-next-generation-getting-started/tutorial/index.html

# Example of hierarchical requirements with links

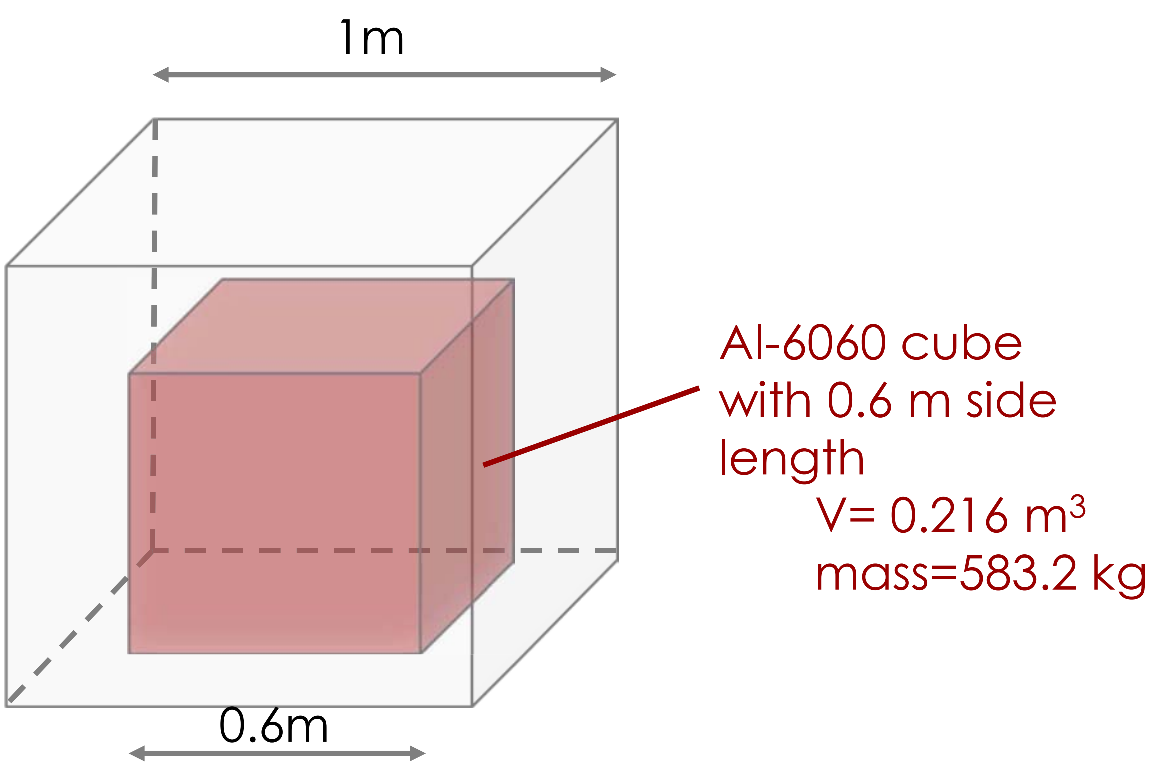
- R1.0 The system shall fit into a volume not exceeding 1.0 m^3
  - R1.1 The system width shall be between 0.5m and 1.0m
  - R1.2 The system height shall be between 0.5m and 1.0m
  - R1.3 The system depth shall be between 0.5m and 1.0m
- R2.0 The system shall be made entirely from Aluminum 6060 alloy
  - R2.1 The system shall not contain any internal voids or cavities
- R3.0 The shape of the system must be a cube
  - R3.1 The angles between sides shall be 90 deg +/-1 deg
  - R3.2 etc…

*Embedding hyperlinks in a requirements document, helps with Traceability. Links can be one-to-one or many-to-one.*

# Derived requirement (linked)

- R4.0 The mass of the system shall not exceed 2,700 kg
  - R4.1 The center of mass of the system must be located at least 0.25 meters from the edge of its volumetric envelope
  - R4.2 The mass of the system must be verified using a Mettler-Toledo XYZ Bench Scale

*What kind of object would satisfy these requirements?*



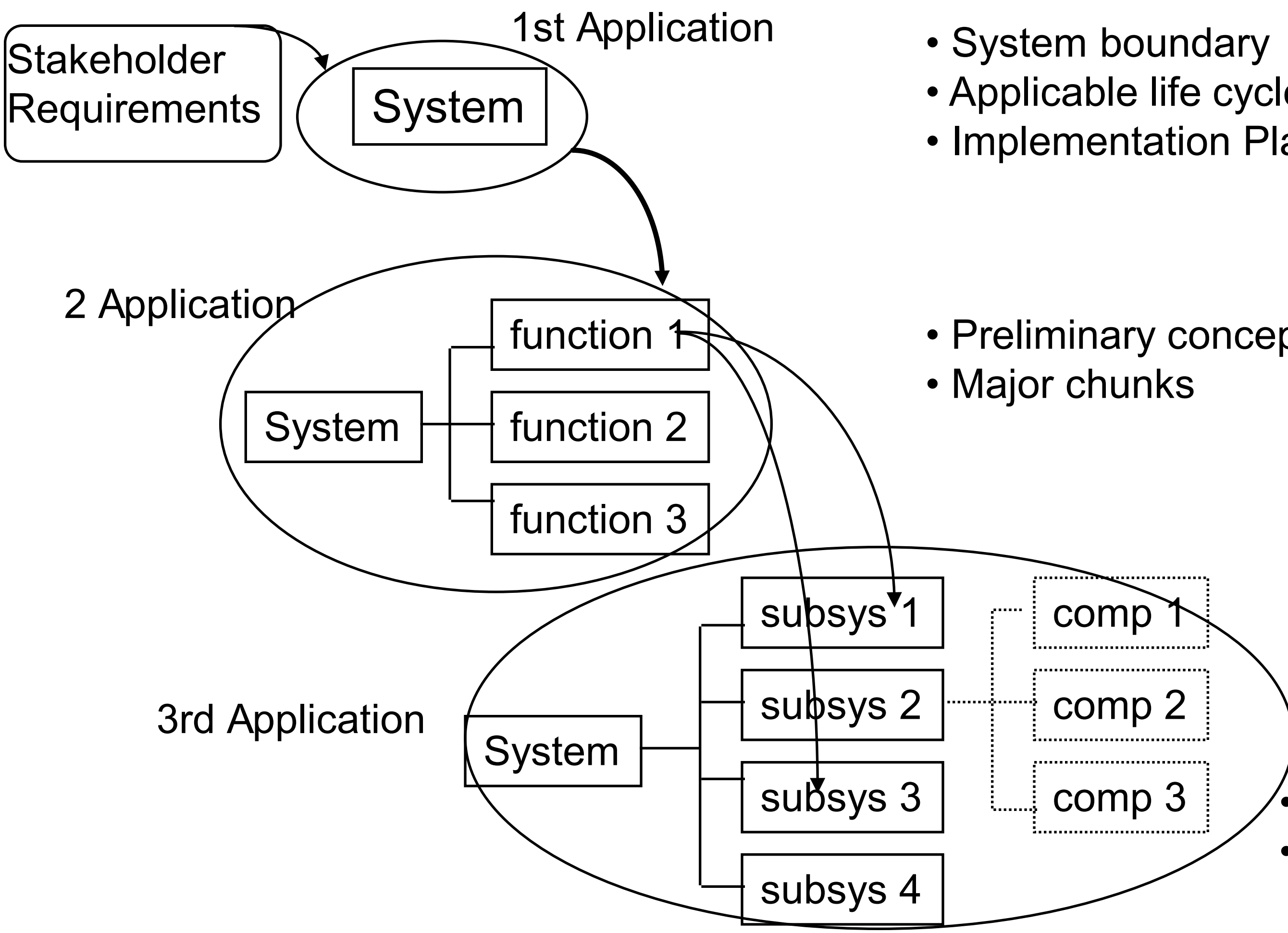
# Overview

- What are requirements?
  - Definition, Examples, Evolution, Standards
- NASA Requirements Process
- Challenges of Requirements Definition
  - Flowdown and Allocation → Isoperformance
  - Validation and Verification
  - Writing good requirements
- What happens at the SRR?
- Kickoff Assignment A2

# Requirements Allocation

- Decompose system requirements into lower levels of design.
  - Define all the lower level functions which must be performed to satisfy the requirement
  - Create architecture of sub-components to provide those functions
- Allocate a level of performance to each lower level function
  - Specify interface requirements to other sub-systems
- Closure - Ensure that satisfaction of the set of requirements at the lower level will guarantee satisfaction of the higher level requirement.

# Requirements Allocation Process




- System boundary
- Applicable life cycle
- Implementation Plan - top level

- Preliminary concept (function to form)
- Major chunks

- System key performance parameters
- Subsystems defined
- Notional components

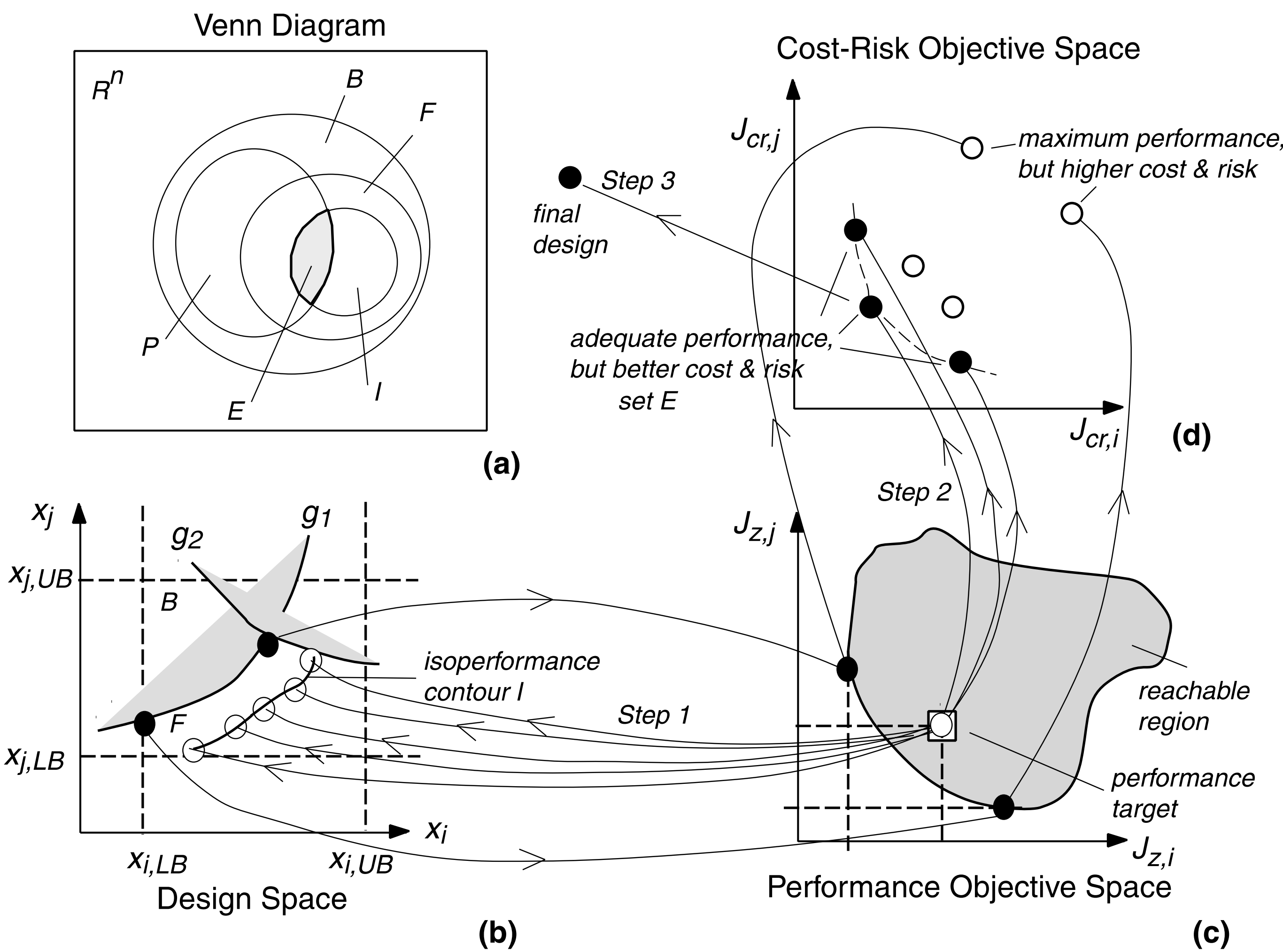
Difficult to decompose reqts to lower levels while staying solution neutral

# Isoperformance Methodology

- "Isoperformance" is a methodology that helps to better do requirements flow-down and allocation
- Starts with a vector of desired performance targets/requirements
- Runs simulation models of systems to determine if
  - A) the vector of targets is feasible
  - B) and if so, produce a set of non-unique feasible combinations of designs to establish correct sub-system requirements

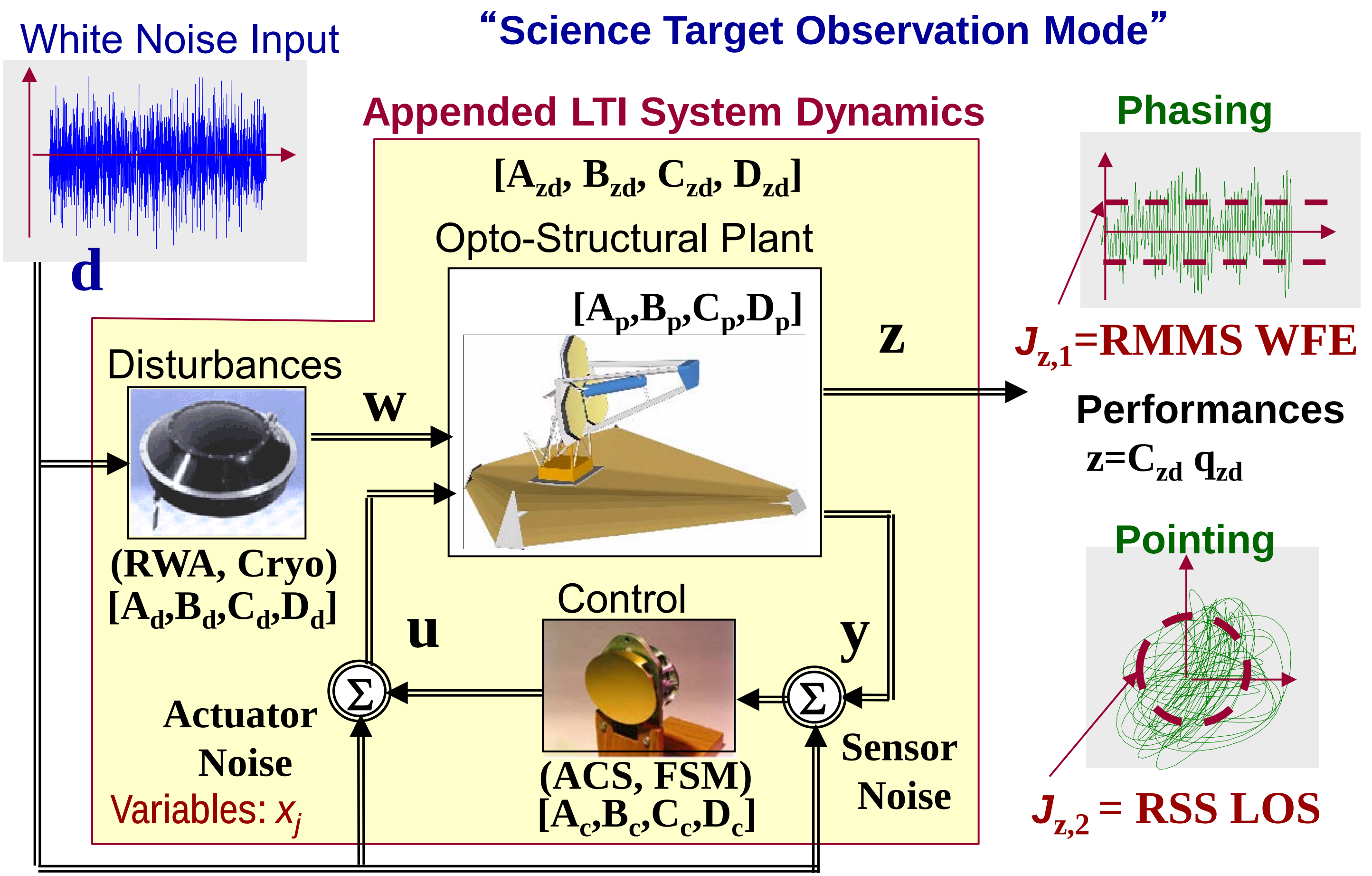
Reading [2b]: de Weck, O.L. and Jones M. B., "Isoperformance: Analysis and Design of Complex Systems with Desired Outcomes", *Systems Engineering*, 9 (1), 45-61, January 2006 (17 pages)

# Requirements and Isoperformance




© Wiley Periodicals, Inc. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

# Isoperformance for Requirements Allocation




© MIT. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

# Example: Nexus Case Study

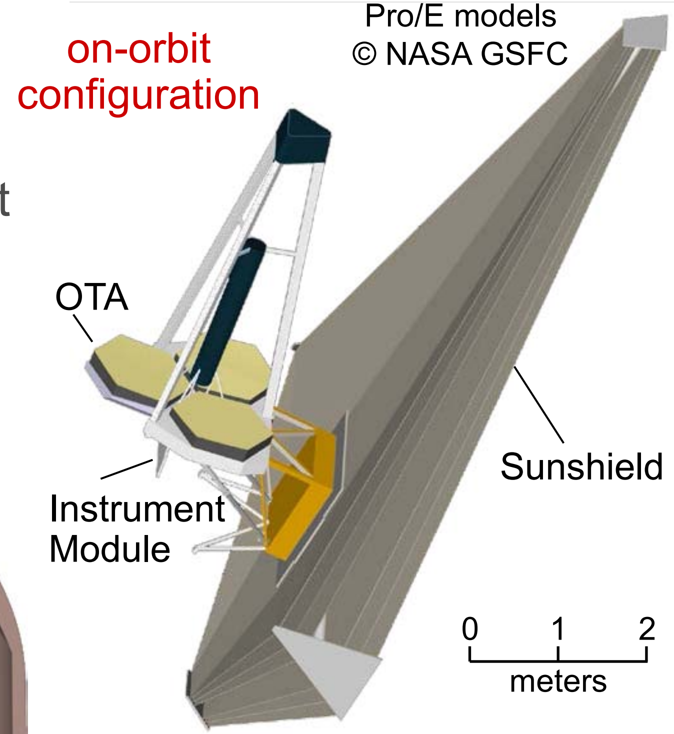
Purpose of this case study:

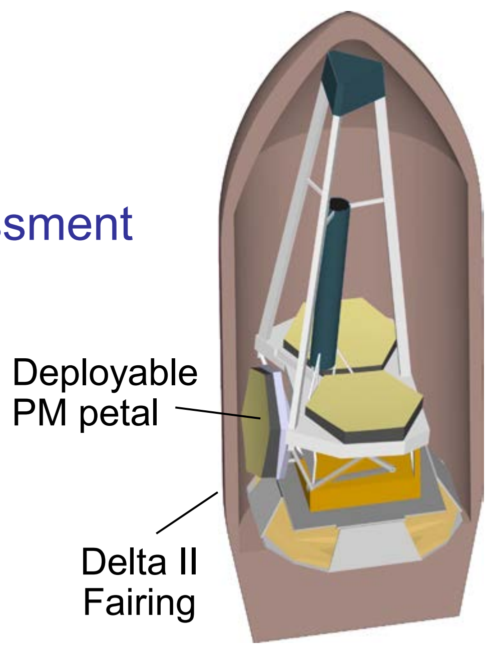
Demonstrate the usefulness of Isoperformance on a realistic conceptual design model of a high-performance spacecraft

The following results are shown:

- Integrated Modeling
- Nexus Block Diagram
- Baseline Performance Assessment
- Sensitivity Analysis
- Isoperformance Analysis (2)
- Multiobjective Optimization
- Error Budgeting

Nexus Spacecraft Concept



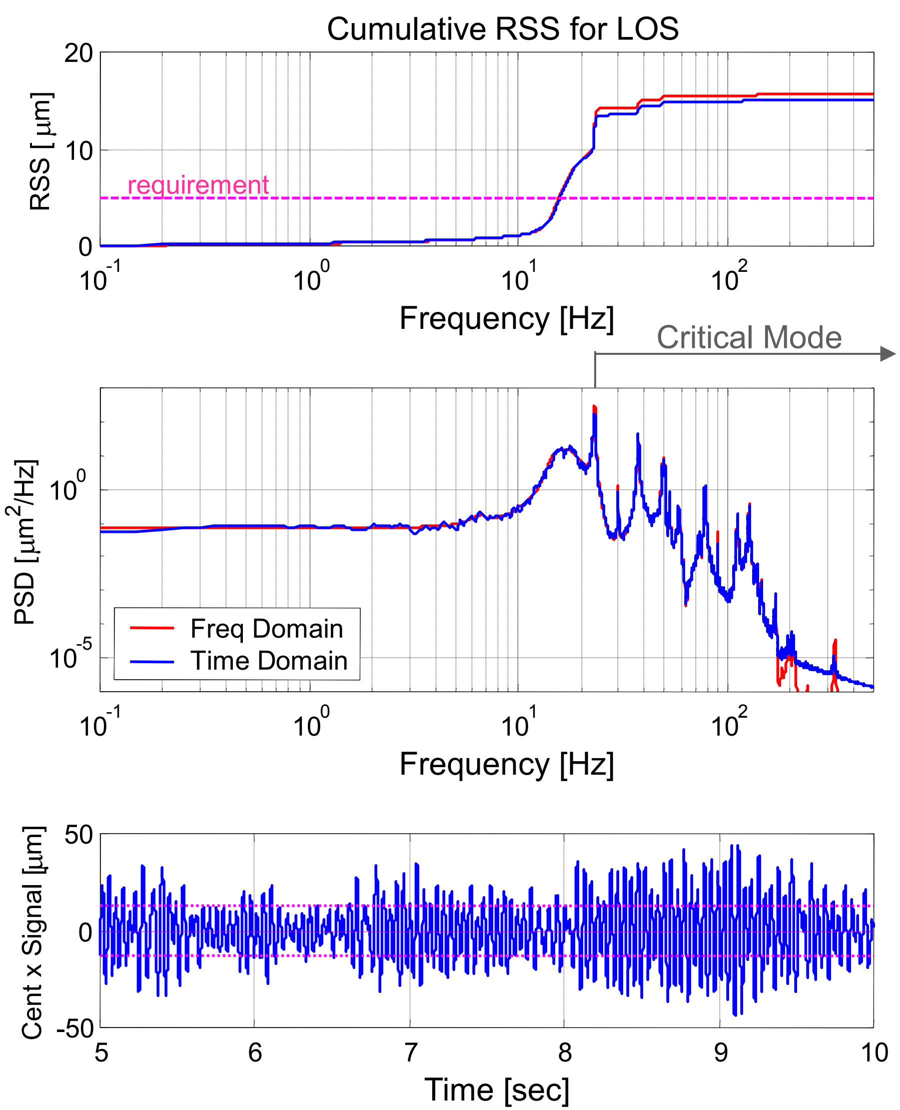



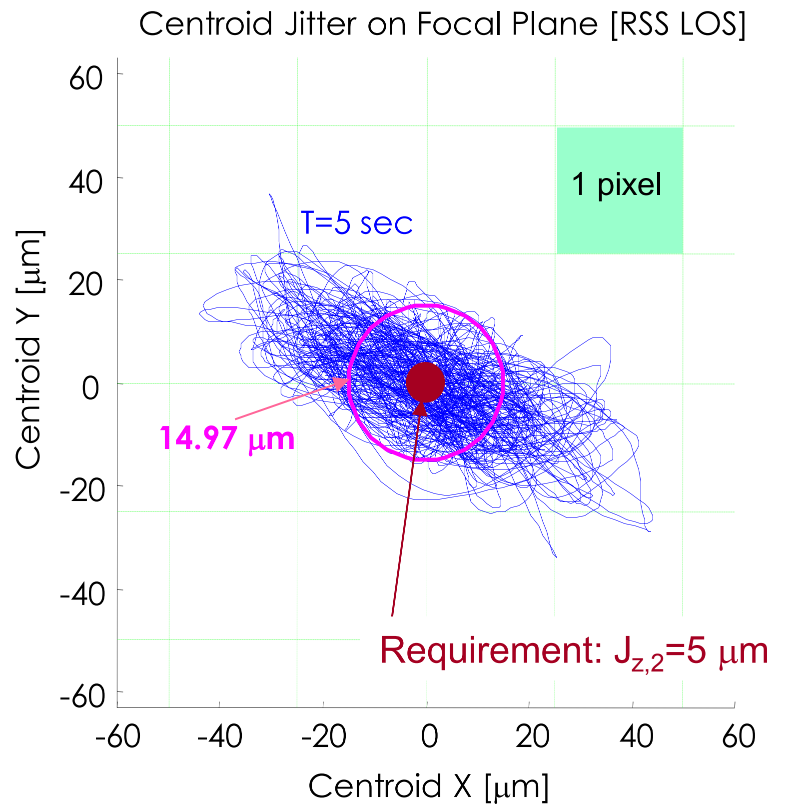

NGST Precursor Mission
2.8 m diameter aperture
Mass: 752.5 kg
Cost: 105.88 M$ (FY00)
Target Orbit: L2 Sun/Earth
Projected Launch: 2004

© MIT. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

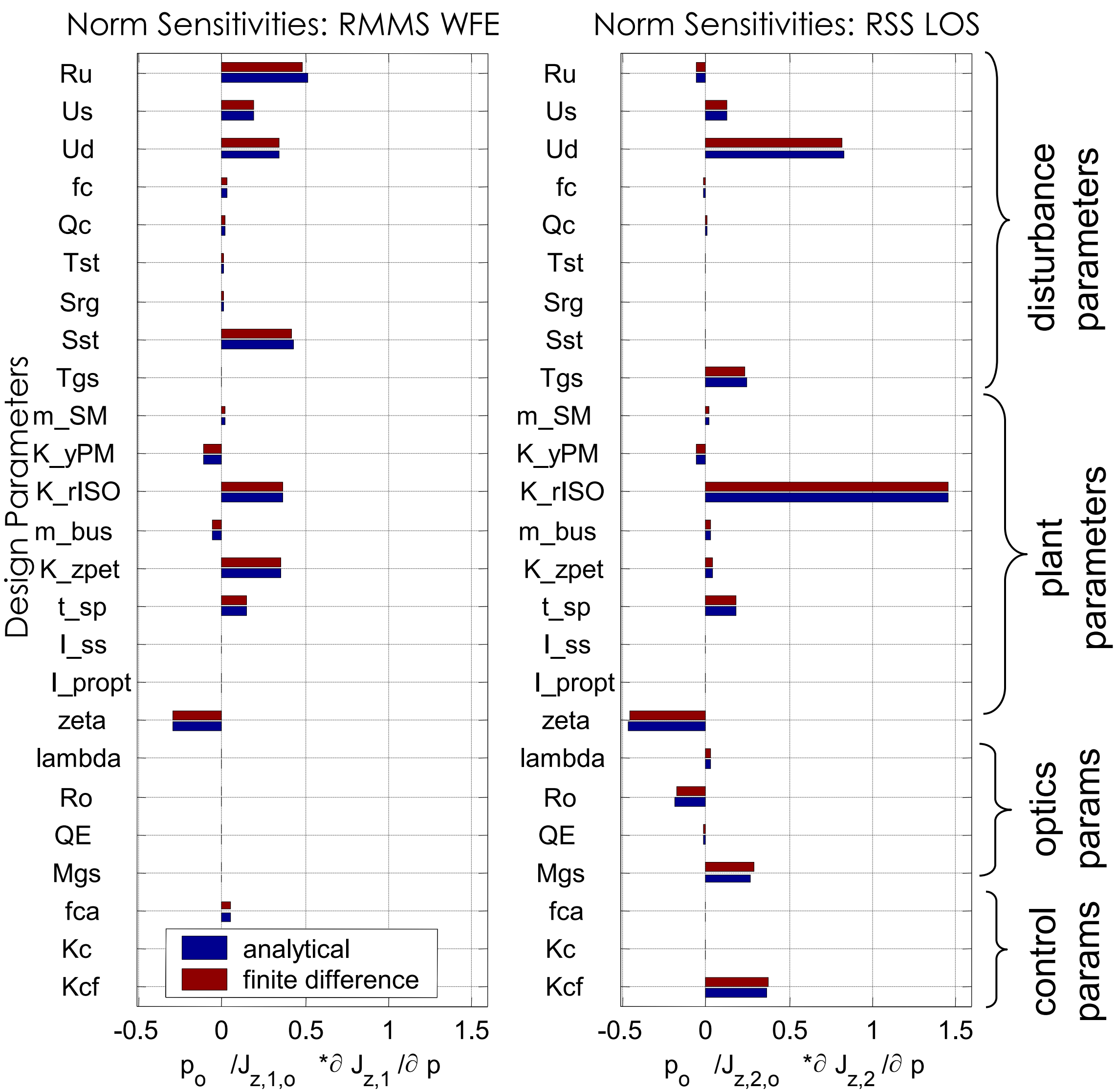
# Initial Performance Assessment $J_z(p^o)$

| Results | Lyap/Freq | Time | |
|---|---|---|---|
| $J_{z,1}$ (RMMS WFE) | **25.61** | **19.51** | [nm] |
| $J_{z,2}$ (RSS LOS) | **15.51** | **14.97** | [μm] |



# Nexus Sensitivity Analysis




Graphical Representation of Jacobian evaluated at design $p_o$, normalized for comparison.

$$\overline{\nabla}J_z = \frac{p_o}{J_{z,o}} \begin{bmatrix} \frac{\partial J_{z,1}}{\partial R_u} & \frac{\partial J_{z,2}}{\partial R_u} \\ \cdots & \cdots \\ \frac{\partial J_{z,1}}{\partial K_{cf}} & \frac{\partial J_{z,2}}{\partial K_{cf}} \end{bmatrix}$$

**RMMS WFE most sensitive to:**

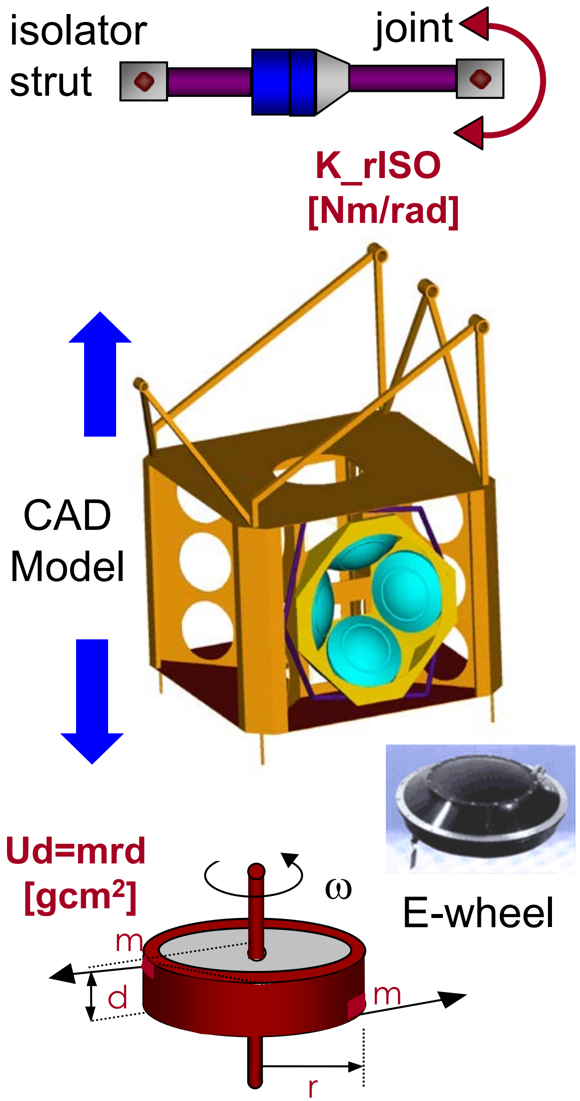
Ru - upper op wheel speed [RPM]
Sst - star track noise 1σ [asec]
K_rISO - isolator joint stiffness [Nm/rad]
K_zpet - deploy petal stiffness [N/m]

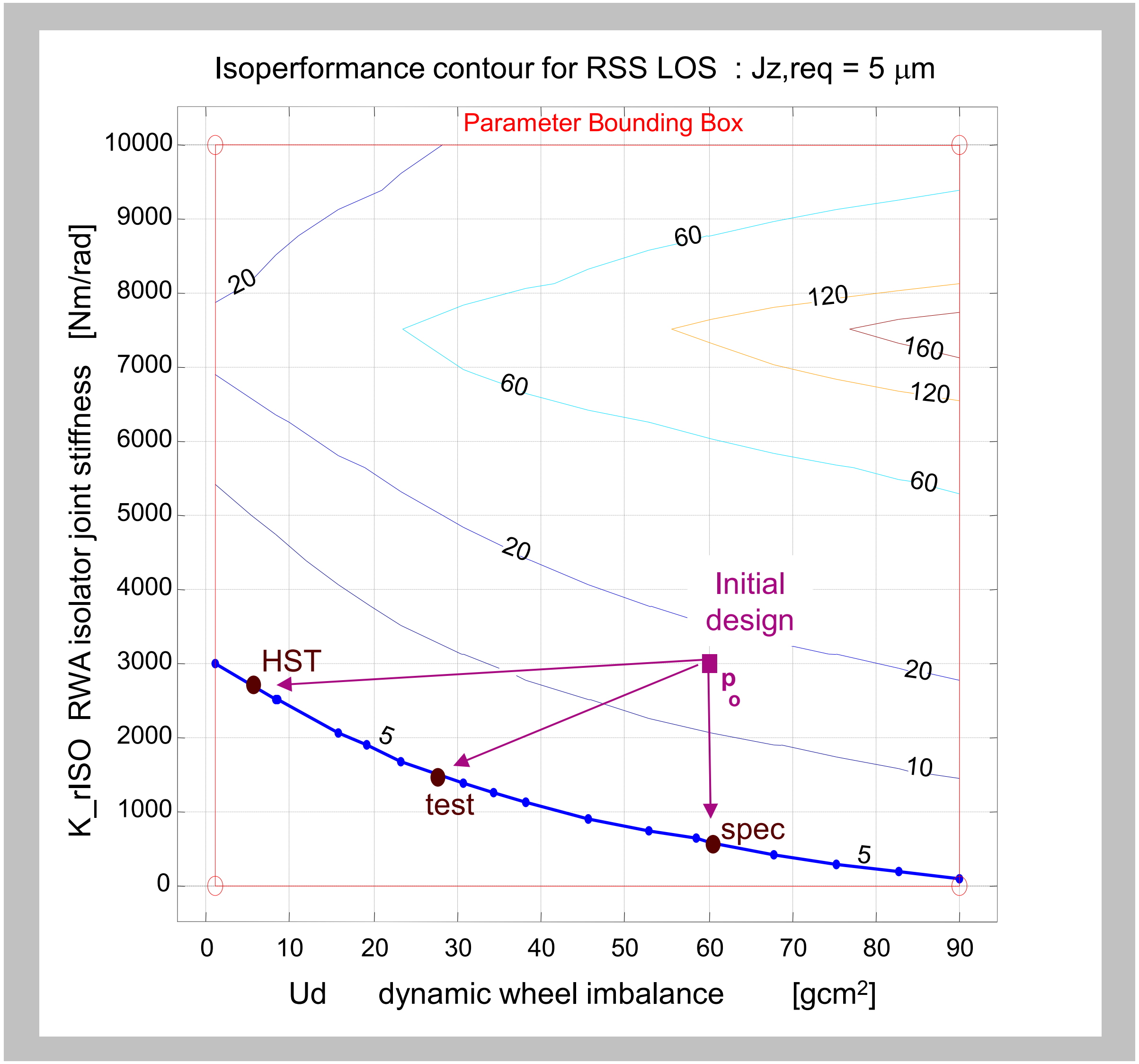
**RSS LOS most sensitive to:**

Ud - dynamic wheel imbalance [$gcm^2$]
K_rISO - isolator joint stiffness [Nm/rad]
zeta - proportional damping ratio [-]
Mgs - guide star magnitude [mag]
Kcf - FSM controller gain [-]

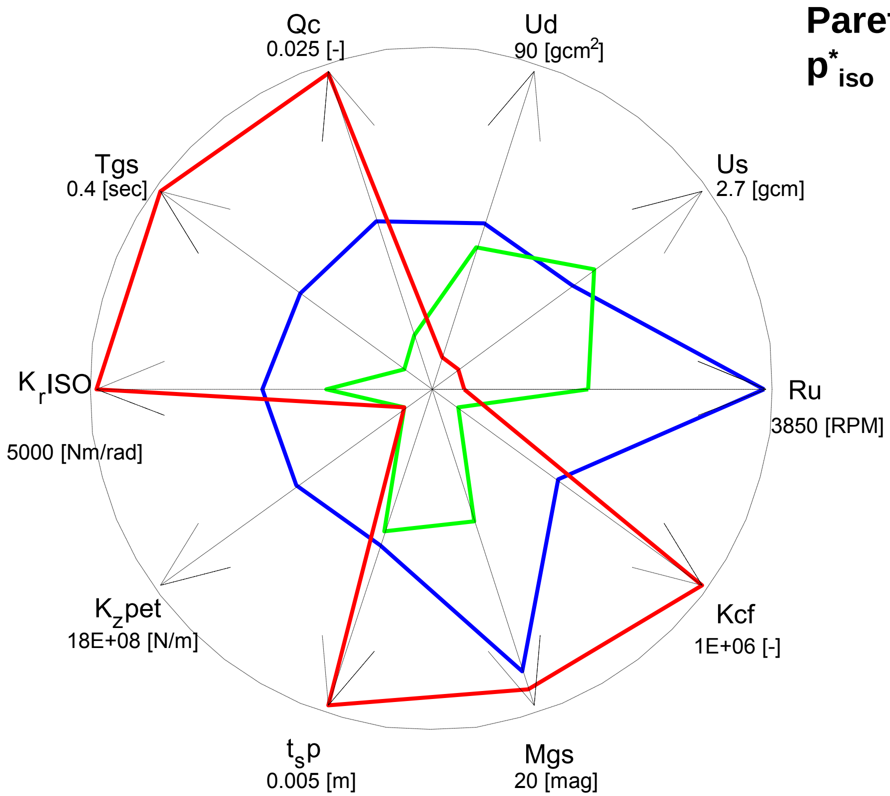
© Massachusetts Institute of Technology - Prof. de Weck - 2015

# 2D-Isoperformance Analysis



# Nexus Multivariable Isoperformance $n_p$=10




**Pareto-Optimal Designs**

**$p^*_{iso}$**

Design A

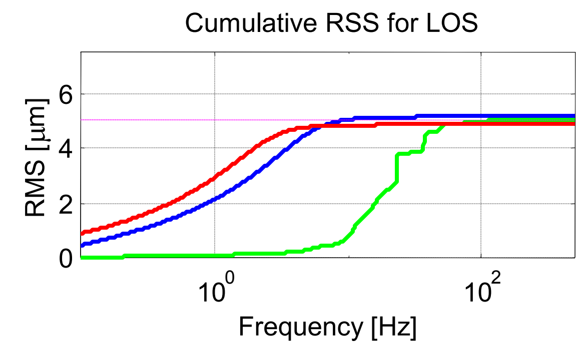
Best "mid-range" compromise

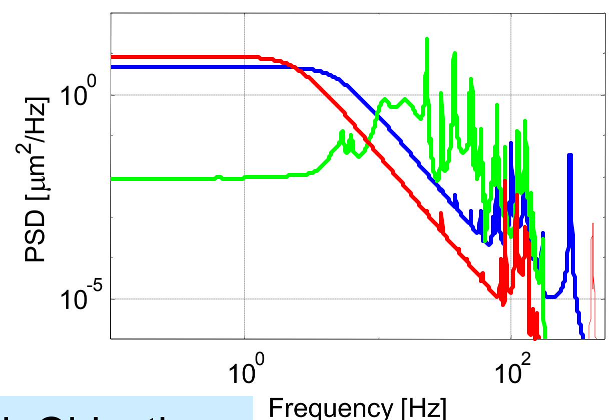
Design B

Smallest FSM control gain

Design C

Smallest performance uncertainty

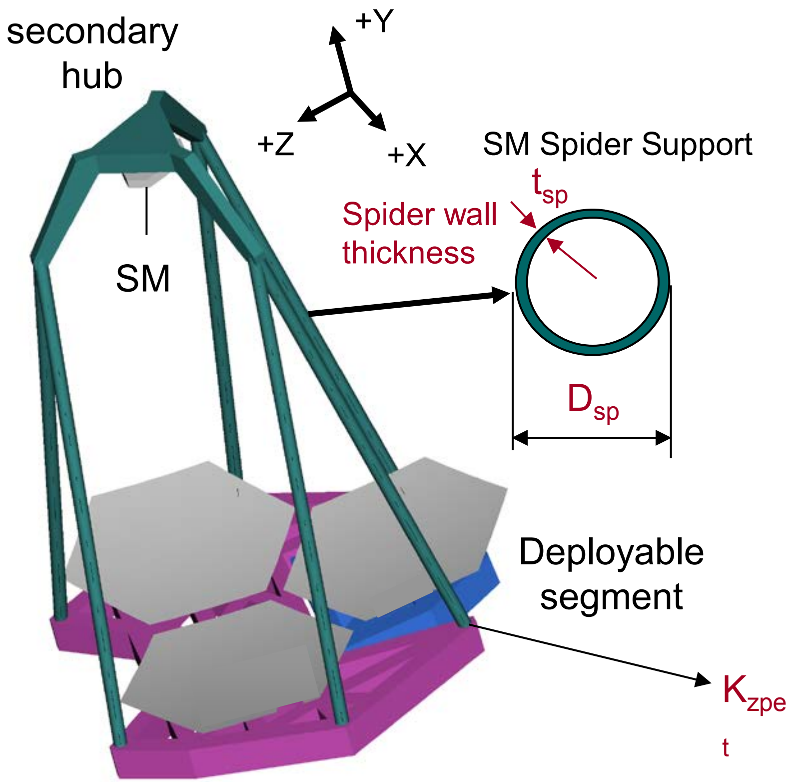


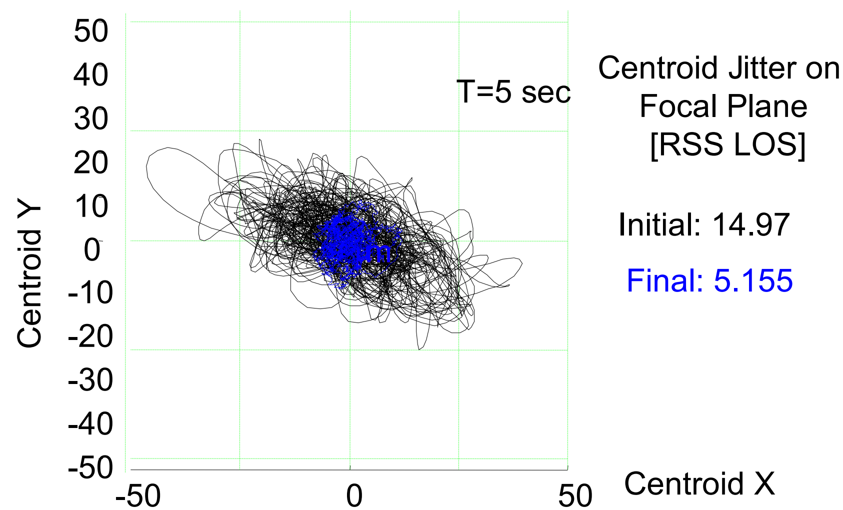

A: min($J_{c1}$)
B: min($J_{c2}$)
C: min($J_{r1}$)

| | Performance | | Cost and Risk Objectives | | |
|---|---|---|---|---|---|
| | Jz,1 | Jz,2 | Jc,1 | Jc,2 | Jr,1 |
| Design A | 20.0000 | 5.2013 | 0.6324 | 0.4668 | +/- 14.3218 % |
| Design B | 20.0012 | 5.0253 | 0.8960 | 0.0017 | +/- 8.7883 % |
| Design C | 20.0001 | 4.8559 | 1.5627 | 1.0000 | +/- 5.3067 % |

# Nexus Initial $p^o$ vs. Final Design $p^{**}_{iso}$




| Parameters | Initial | Final |
|---|---|---|
| Ru | 3000 | 3845[RPM] |
| Us | 1.8 | 1.45 [gcm] |
| Ud | 60 | 47.2 [gcm$^2$] |
| Qc | 0.005 | 0.014 [-] |
| Tgs | 0.040 | 0.196 [sec] |
| KrISO | 3000 | 2546[Nm/rad] |
| Kzpet | 0.9E+8 | 8.9E+8 [N/m] |
| tsp | 0.003 | 0.003 [m] |
| Mgs | 15 | 18.6 [Mag] |
| Kcf | 2E+3 | 4.7E+5 [-] |




**Improvements are achieved by a well balanced mix of changes in the disturbance parameters, structural redesign and increase in control gain of the FSM fine pointing loop.**

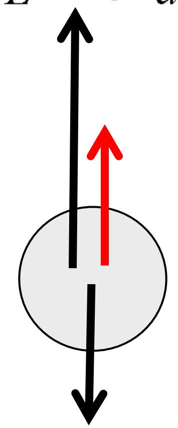
# Concept Question 4

- **A balloon shall lift a payload of 1,000 kilograms, including its own mass**. It can use either helium ($\rho_{He}$=0.2 kg/m$^3$) or hydrogen ($\rho_H$=0.1 kg/m$^3$) as a lift gas. The standard density of air is $\rho_{Air}$=1.3 kg/m$^3$.
- $r_b$= radius of balloon, $m_b$ = mass of balloon
- Which of the following requirements is *infeasible*?
  - A- The balloon shall have a radius of 6.1 meters. The balloon shall use 99.9% pure helium as a lift gas.
  - B – The balloon shall have a radius of 5.9 meters. The balloon shall use 99.9% pure hydrogen as a lift gas.
  - C – The balloon shall have a radius of 5.9 meters. The balloon shall use 99.9% helium as a lift gas.
  - D – The balloon shall have a radius of 6.1 meters. The balloon shall use 99.9% hydrogen as a lift gas.
  - E – All these requirements are okay.
  - F – None of these requirements are feasible.

- Answer Concept Question 4 (see supplemental files)

# Concept Question 4 - Solution

$$F_L = \rho_{air} V_b g$$



$$F_W = \left(m_b + \rho_{He} V_b\right) g$$

For lift need $F_R >= 0$

$$V_b \geq \frac{m_b}{\rho_{air} - \rho_{gas}}$$

Lift Condition

$$\rho_{He} = 0.2 \ [\mathrm{kg/m^3}]$$
$$\rho_{H} = 0.1 \ [\mathrm{kg/m^3}]$$
$$\rho_{air} = 1.3 \ [\mathrm{kg/m^3}]$$

$$g = 10 \ [\mathrm{m/s^2}]$$

$$V_b = \frac{4}{3}\pi r_b^3$$

**Solution**

| Case | Gas | $r_b$ | $F_R>0$ | Feasible |
|---|---|---|---|---|
| A | He | 6.1 m | yes | yes |
| B | H2 | 5.9 m | yes | yes |
| **C** | **He** | **5.9 m** | **no** | **no** |
| D | H2 | 6.1 m | yes | yes |

# Common problems with requirements

- Writing implementations ("How") instead of requirements ("What")
  - Forces the design
  - Implies the requirement is covered
- Using incorrect terms
  - Avoid "support", "but not limited to", "etc", "and/or"
- Using incorrect sentence structure or bad grammar

# Common problems continued

- Writing unverifiable requirements
  - E.g., minimize, maximize, rapid, user-friendly, easy, sufficient, adequate, quick
- Missing requirements
  - Requirement drivers include

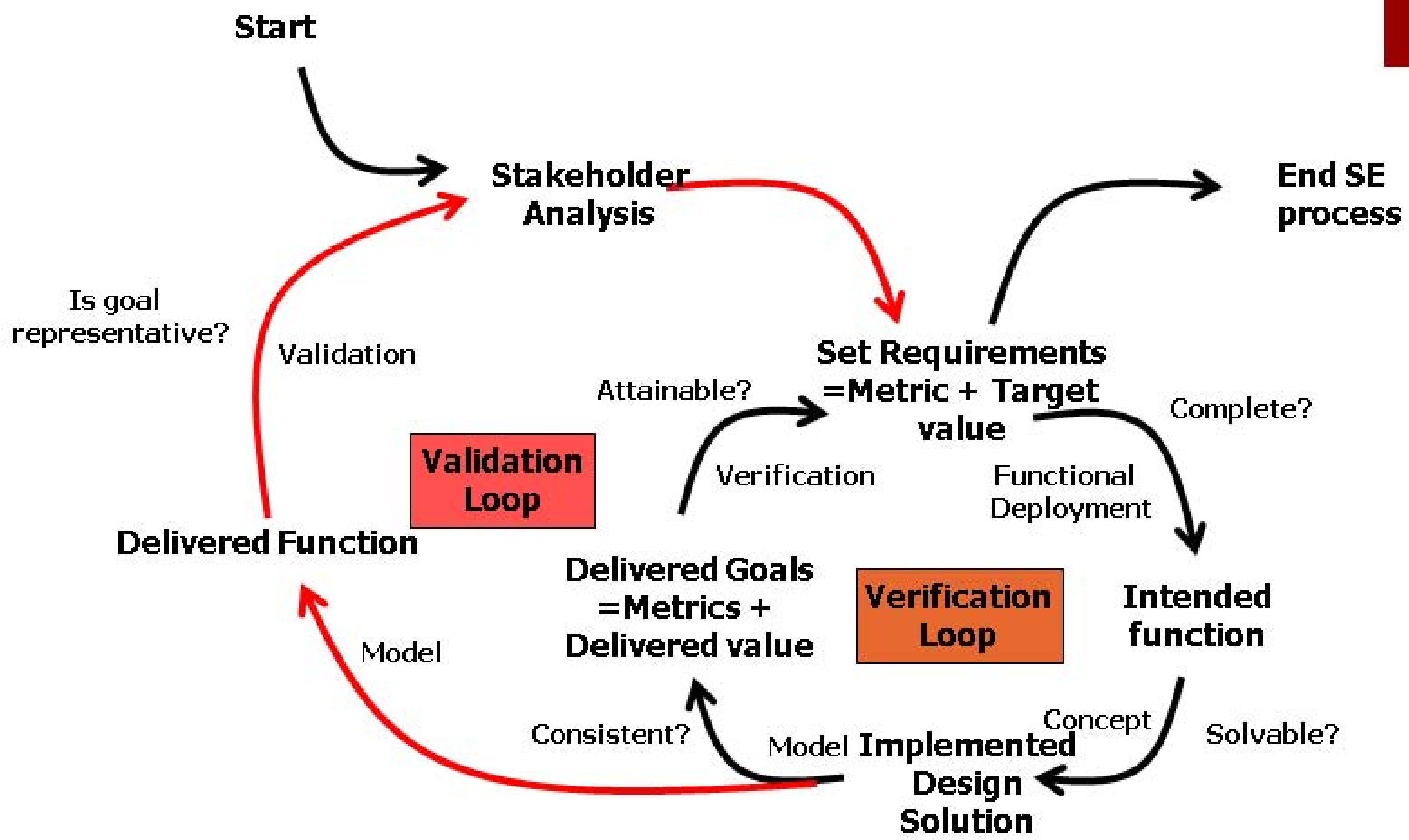
| | | |
|---|---|---|
| Functional | Performance | Interface |
| Environment | Facility | Transportation |
| Training | Personnel | Reliability |
| Maintainability | Operability | Safety |

- Requirements only written for "first use"
- Over-specifying

# Verification

- Every requirement must be verified to ensure that the proposed design actually satisfies the requirement by
  - Examination,
  - Test,
  - Demonstration, or
  - Analysis
- Requirements documentation specifies the development phase and method of verification

# Verification and Validation Loops



# Overview

- What are requirements?
  - Definition, Examples, Evolution, Standards
- NASA Requirements Process
- Challenges of Requirements Definition
  - Flowdown and Allocation → Isoperformance
  - Validation and Verification
  - Writing good requirements
- What happens at the SRR?
- Kickoff Assignment A2

# System Requirements Review (SRR)

- SRR is primarily a human / social peer-review process
- Main goal of SRR: **Vet the requirements as written**. Find any missing, misstated, redundant or otherwise unsatisfactory requirements.

*Richard Kornfeld*

This image is in the public domain.

Image Source: http://www.jpl.nasa.gov/images/msl/20120821b/mslgroup-640.jpg

# What happens at SRR ?

- **SRR** typically occurs rather early in a program, during Phase A (Concept and Technology Development) and before Phase B (Preliminary Design)
- Typically need to have the top two levels of requirements (L0 = mission requirement / CONOPS), that is L1 (major systems) and L2 (major subsystems) clearly defined
- Requirements need to be written down, validated and accessible to everyone on the team (document or database)
- Requirements are put under formal configuration management (=version control) at this point. Adding, deleting or modifying requirements requires management approval. Systems Engineers are very involved.

**Table 6.7-1 Program Technical Reviews**

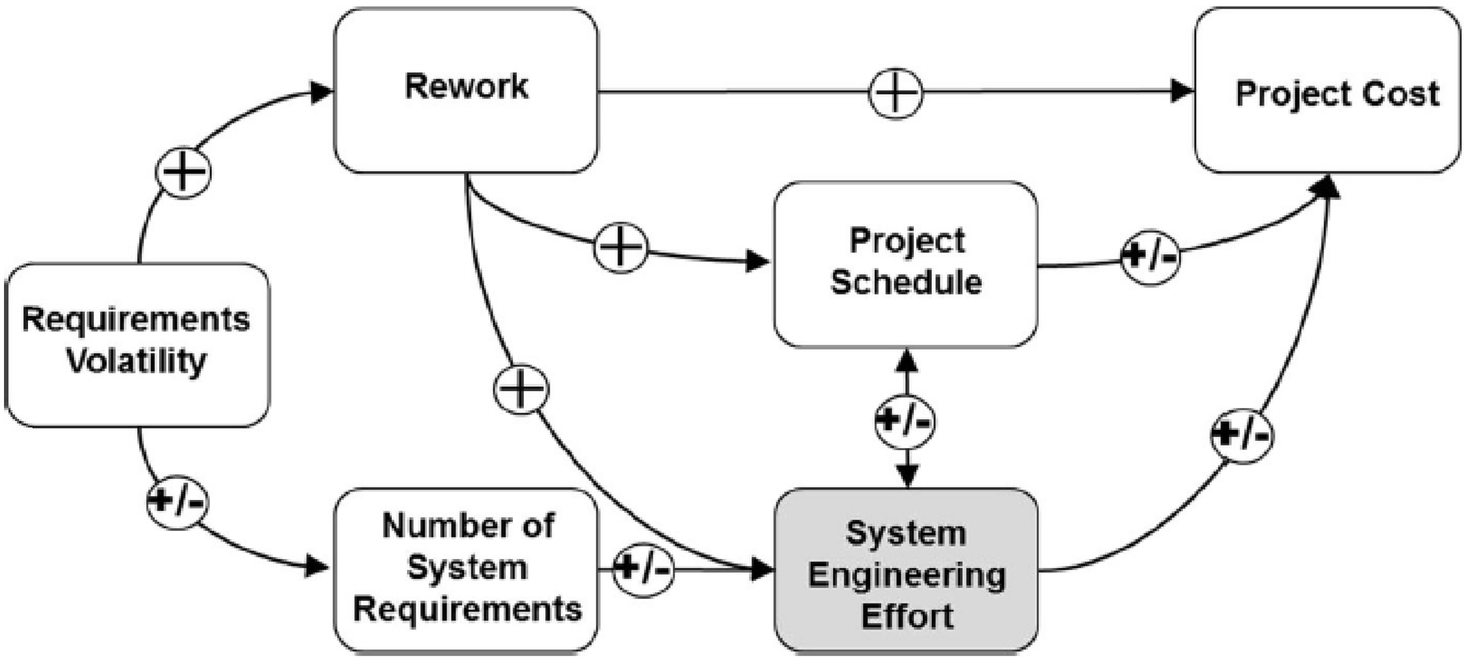
| Review | Purpose |
|---|---|
| Program/ System Requirements Review | The P/SRR examines the functional and performance requirements defined for the program (and its constituent projects) and ensures that the requirements and the selected concept will satisfy the program and higher level requirements. It is an internal review. Rough order of magnitude budgets and schedules are presented. |
| Program/ System Definition Review | The P/SDR examines the proposed program architecture and the flowdown to the functional elements of the system. |

This image is in the public domain.

NASA SE Handbook, p.170

# Requirements Volatility

- **Requirements Volatility** is the degree to which requirements continue to change during a project, even post-SRR
  - % of addition, removal, modification of requirements per time period
  - Variety of causes of requirements volatility (see below), generally undesirable




© John Wiley and Sons. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/help/faq-fair-use/.

Reading [2a]: Peña, M. and Valerdi, R. (2015), Characterizing the Impact of Requirements Volatility on Systems Engineering Effort. *Systems Engineering*, 18: 59–70. doi: 10.1111/sys.21288

# Overview

- What are requirements?
  - Definition, Examples, Evolution, Standards
- NASA Requirements Process
- Challenges of Requirements Definition
  - Flowdown and Allocation → Isoperformance
  - Validation and Verification
  - Writing good requirements
- What happens at the SRR?
- Kickoff Assignment A2

# Assignment A2 Kickoff

| *Assignment* | *Topic* | *Weight* |
|---|---|---|
| A1 (group) | Team Formation, Definitions, Stakeholders, Concept of Operations (CONOPS) | 12.5% |
| A2 (group) | Requirements Definition and Analysis<br>Margins Allocation | 12.5% |
| A3 (group) | System Architecture, Concept Generation | 12.5% |
| A4 (group) | Tradespace Exploration, Concept Selection | 12.5% |
| A5 (group) | Preliminary Design Review (PDR) Package and Presentation | 20% |
| Quiz (individual) | Written online quiz | 10% |
| Oral Exam (individual) | 20' Oral Exam with Instructor<br>2-page reflective memorandum | 10% |

# Assignment A2

- Out today
- **Due in 2 weeks**
- Weight: 1/8 (12.5% of your grade)
- Same teams as did Assignment 1 (max number of 5 team members)
- Tasks
  1. Find a project/program where poorly written or managed requirements were a major problem. Discuss this case as a team. (20%)
  2. Analyze the requirements as written in the Cansat 2016 Mission Guide (47 base requirements) and the Environmental Testing Manual (30%)
  3. Generate your own set of requirements for the Cansat 2016 competition in the form of a requirements document or database (30%)
  4. Allocate/flow down requirements to subsystems (without selecting a specific design concept) and allocate margins (=reserves). (20%)
- Upload your responses to stellar / moodle as appropriate

# Summary of Session 2

- Good Requirements are essential for driving system design
  - Basis for cost and schedule estimation
  - Sets up necessary verification and validation activities
- Requirements flow-down is challenging
  - Level 0 / Level 1 Requirements from Stakeholder analysis and ConOps
  - Level 2, 3 etc... requirements emerge later during Preliminary Design
- Some methods and also commercial tools exist for formal requirements management
  - Isoperformance → requirements allocation given upper level targets
  - DOORS → part of IBM Rational Suite. Professional requirements management
- Requirements Volatility
  - Purpose of SRR is to review and approve high level requirements
  - Even later requirements can still change → rework impacts project schedule/cost

MIT OpenCourseWare
http://ocw.mit.edu

16.842 Fundamentals of Systems Engineering
Fall 2015

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.