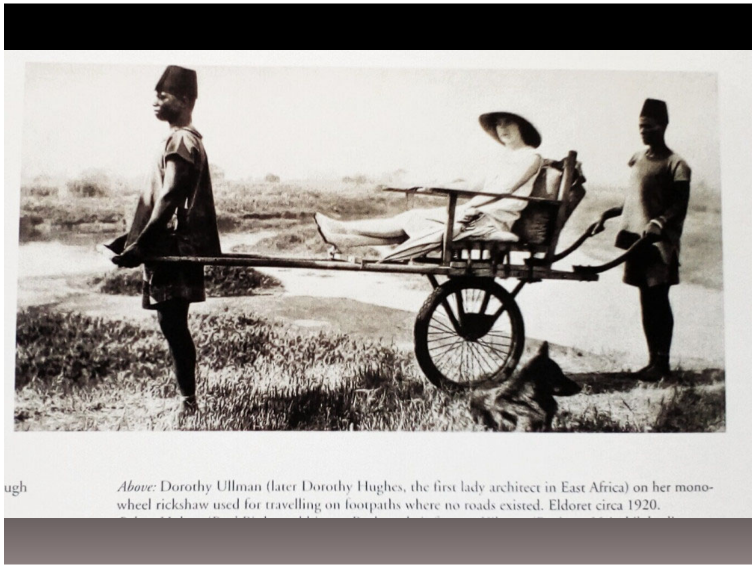
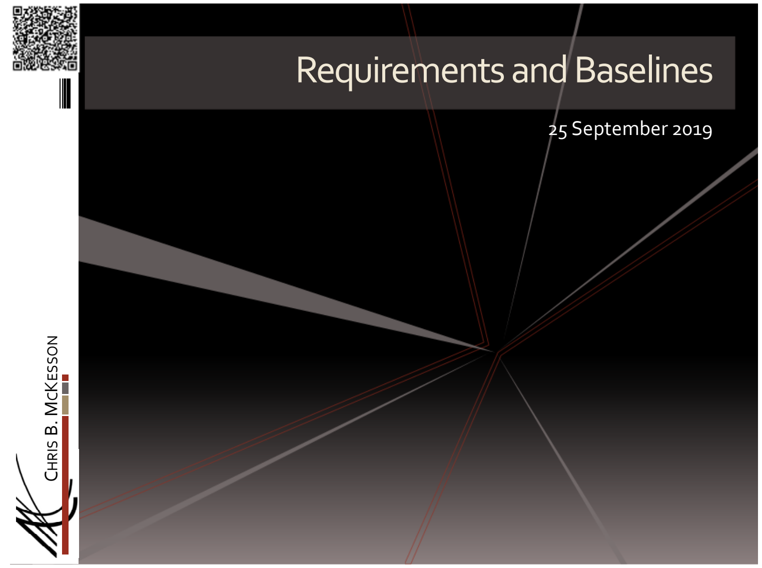
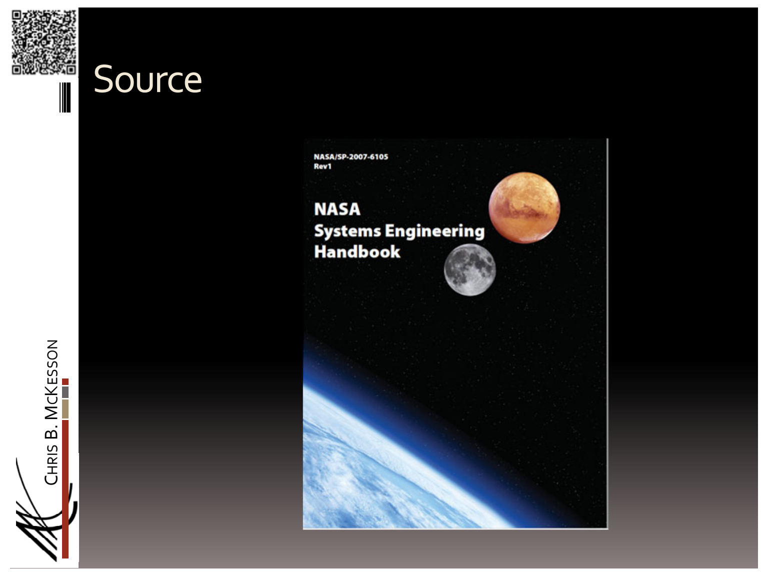
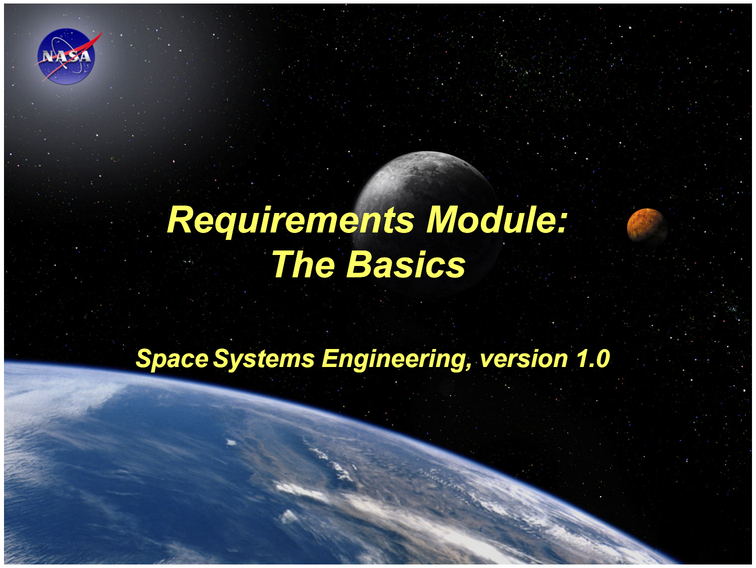

Above: Dorothy Ullman (later Dorothy Hughes, the first lady architect in East Africa) on her mono-wheel rickshaw used for travelling on footpaths where no roads existed. Eldoret circa 1920.


Requirements and Baselines
25 September 2019
Chris B. McKesson


Requirements and Baselines
25 September 2019
DON'T FIX IT!
Chris B. McKesson


Source
NASA/SP-2007-6105
Rev1
NASA
Systems Engineering
Handbook
Chris B. McKesson



Module Purpose: Requirements - The Basics

- Establish the role of good requirements in project success.
  - Requirements capture the understanding of what is to be done.
- Establish the significance of good requirements development.
  - Poor requirements are the single biggest problem for projects.
  - The later a problem is discovered the more costly it is to recover from.
- Describe the different types of requirements.
- Establish how requirements are distributed — allocation, flow-down and derived.
- Define and understand the value of requirements traceability.
- Recognize that system decomposition creates new interfaces that must be defined and owned.

Module Purpose: Requirements - The Basics

- Establish the role of good requirements in project success.
  - Requirements capture the understanding of what is to be done.
- Establish the significance of good requirements development.
  - Poor requirements are the single biggest problem for projects.
  - The later a problem is discovered the more costly it is to recover from.
- Describe the different types of requirements.
- Establish how requirements are distributed — allocation, flow-down and derived.
- Define and understand the value of requirements traceability.
- Recognize that system decomposition creates new interfaces that must be defined and owned.

Module Purpose: Requirements - The Basics

- Establish the role of good requirements in project success.
  - Requirements capture the understanding of what is to be done.
- Establish the significance of good requirements development.
  - Poor requirements are the single biggest problem for projects.
  - The later a problem is discovered the more costly it is to recover from.
- Describe the different types of requirements.
- Establish how requirements are distributed — allocation, flow-down and derived.
- Define and understand the value of requirements traceability.
- Recognize that system decomposition creates new interfaces that must be defined and owned.

Module Purpose: Requirements - The Basics

- Establish the role of good requirements in project success.
  - Requirements capture the understanding of what is to be done.
- Establish the significance of good requirements development.
  - Poor requirements are the single biggest problem for projects.
  - The later a problem is discovered the more costly it is to recover from.
- Describe the different types of requirements.
- Establish how requirements are distributed — allocation, flow-down and derived.
- Define and understand the value of requirements traceability.
- Recognize that system decomposition creates new interfaces that must be defined and owned.

The Importance of Requirements

- Requirements problems are the single biggest cause of project problems.
- Requirements define what is to be done, how well and under what constraints - get the requirements wrong and the design and hardware will be wrong.
- Requirements drive…

  Cost - Design - Schedules - Skills required - Verification plans - Operational procedures - *everything*
- *It is amazing how many teams begin to solve a problem before there is agreement on what the problem is.* Requirements and their associated constraints and assumptions quantify the problem to be solved - they establish how project success will be determined.

The Importance of Requirements

- Requirements problems are the single biggest cause of project problems.
- Requirements define what is to be done, how well and under what constraints - get the requirements wrong and the design and hardware will be wrong.
- Requirements drive…

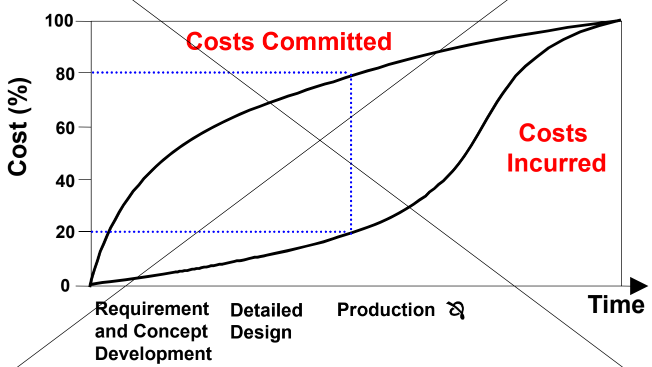
  Cost - Design - Schedules - Skills required - Verification plans - Operational procedures - *everything*
- *It is amazing how many teams begin to solve a problem before there is agreement on what the problem is.* Requirements and their associated constraints and assumptions quantify the problem to be solved - they establish how project success will be determined.

What is a Requirement?

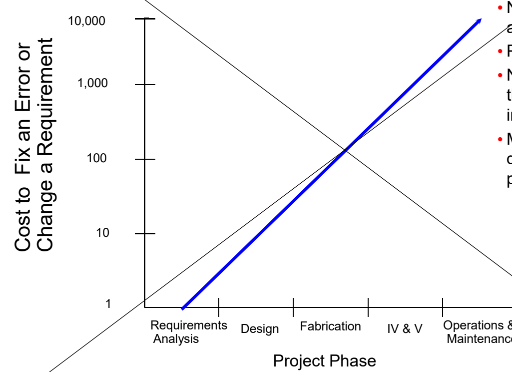
- Statement of some THING you want or need
  OR
  A characteristic of some THING you want or need
- A requirement is also…
  - A *Contractually Binding* Statement
  - Documentation of *Problem Space*
  - The *Means* We Use to Communicate

The Importance of Getting the Requirements Right

Requirement and concept development commit costs before they are incurred.



The Cost of Error Recovery or a Requirements Change Increases Dramatically with Project Phase



Why requirements change:

- New requirements are added or discovered
- Priorities change
- New understanding of the difficulties of an implementation approach
- Measured performance does not meet required performance

Where Do Requirements Come From?

At the highest level, from your scoping exercise

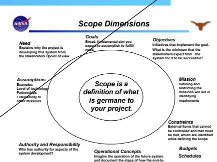
- Stakeholder and Customer need statement
- Defined goals and objectives
- Assumptions and constraints
- Concept of Operations

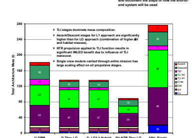
Project analysis and trade studies

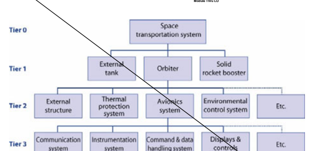
- Figures of merit

Project system hierarchy

- What *functions* must the system and subsystem do to perform the mission?







Requirements Also Come From Organizational Standards and Government Regulations

Guiding Documents, such as

- NASA Procedural Requirements
  - NPR 8020.12C “Planetary Protection Provisions for Robotic Extraterrestrial Missions”
  - NPR 8705.2A “Human-rating Requirements for Space Systems”
- Launch Vehicle payload/user requirements
- Standards-based Requirements
- Regulation Requirements (e.g. DOE nuclear standards)

Further operational considerations

- System boundaries and external *interfaces*
  - Are other systems driving some of your design requirements, like interfacing with the International Space Station?
- The operating and supporting environments
  - What requirements does the space environment impose on your system?
- Use of legacy systems
  - What requirements originally designed to?

Requirements Development Is the Most Important Step!

- Requirements are distributed from the broad mission scope into the architecture that defines the project
- Requirements bound the scope of the problems to be solved so we know when we have done well enough
- A hierarchy of traceable requirements ensures that the project is building only what is required, i.e., no frivolous activities
- A hierarchy of negotiated requirements ensures a balanced system design
- Requirements are the basis for the project's verification and validation efforts
  - Poorly written or unverifiable requirements are trouble!

Requirements are Decomposed Following the Functional Architecture

Requirements are decomposed via three methods — flow-down, allocation and derivation.

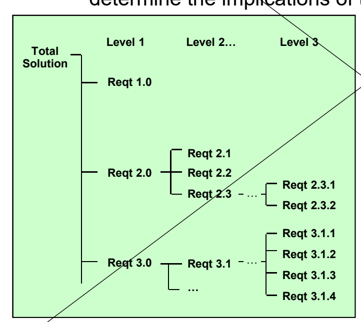
1. Requirement flow-down is a direct transfer since a subsystem provides the capability.
   - E.g., The requirements for spacecraft communications may be entirely flowed-down from the spacecraft system requirements to the spacecraft communications subsystem requirements.
2. Allocation is a quantitative apportionment from a higher level to a lower level and for which the unit of measure remains the same. Examples include mass, power, or pointing.
   - A 1,000 kg spacecraft may allocate 200 kg, 500 kg and 300 kg to its three subsystems.
   - Not always a linear combination - e.g., system pointing performance is typically combined via Root-Summed-Squared (RSS).

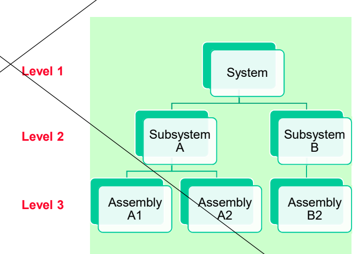
Requirements are Decomposed Following the Functional Architecture

3. Requirement derivation is an apportionment that depends on the specific implementation.
   - E.g., A car may have a 0-100 kph performance requirement that is used to establish a requirement for a maximum mass *and* a minimum horsepower.
   - Or a launch vehicle's performance might establish a maximum satellite mass for a given altitude and orbital inclination.

Requirements Traceability and Hierarchy

- Once mission level requirements have been decomposed to lower levels, traceability identifies the relationship between requirements.
- Knowing the source and dependencies between requirements is valuable since if a requirement changes, traceability can be used to determine the implications of that change.





Types of Requirements

- Functional - Requirements which define ***what*** an item *must do*.
  - The system shall provide communications between the ground and the spacecraft.
- Performance - Requirements which define and quantify ***how well*** an item must accomplish a particular function.
  - Provide communications over what range, with what data rate and how often
- Constraints - Requirements that capture operational, environmental, safety or regulatory constraints.
  - The communications system shall use X-band frequencies.
  - The communications system shall operate with a base plate temperature of at least -30 C and at most 40 C.
  - The maximum RF power density shall be less than 10 watts/m^2
  - Design standards (e.g., metric units, programming language, etc.)
- Verification - Requirements capture how confidence will be established that the system will perform in its intended environment.
  - All performance and functional requirements shall be met while the system is in a vacuum chamber with 2.5 Kwatts/m^2 of visible light illuminating the z-side.

Crew Exploration Vehicle (CEV) Requirements Distribution, example

- System Performance Requirements Document
  - The CEV System shall provide two-way voice communications during crewed operations.
- Flight Segment Performance Requirements Document
  - The flight segment shall provide voice communications to the ground through TDRSS.
  - Other derived requirements.
- Flight Vehicle Contract End Item Specification Part I
  - The flight vehicle shall provide four omni directional antennas.
  - The flight vehicle shall provide two S-Band transponders.
  - The flight vehicle shall provide a switch matrix to allow connection of each S-Band transponder with each omni directional antenna.
  - Other derived requirements.
- Flight Vehicle Contract End Item Specification Part II
  - The communications subsystem shall provide the equipment specified in drawing CEVFV-COM1-234.
  - The communications subsystem shall be wired as shown in drawing CEVFV-COM2-234.
  - The communications subsystem equipment shall be mounted on the avionics pallets as shown in drawing CEVFV-COM2-235.
  - Other derived requirements.

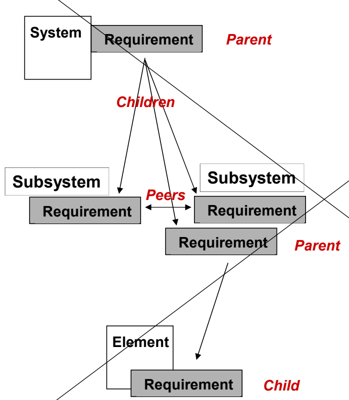
System Functional Performance Requirements

Segment Functional Performance Requirements

Element Design Requirements

Element Fabrication Requirements

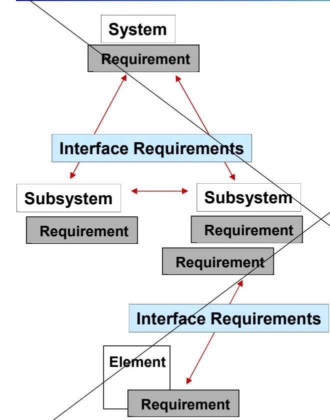
Requirement Families



If parent requirements are…
- INCOMPLETE
- INCORRECT
- AMBIGUOUS
- CONFLICTING, or
- UNVERIFIABLE

Then…children and subsequent generation requirements will be progressively worse.

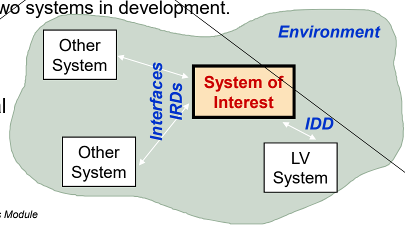
The Decomposition of a System Also Creates Interface Requirements



There can be interfaces between each subsystem and between each subsystem and the system.
The functions, performance, assumptions and constraints of these interfaces much be defined and captured in interface requirements.
Interface ownership must be established - since the responsible individual or organization for an interface is not always obvious.

Key Interface Documents

- Interface Definition Document (IDD) - defines interfaces to an existing system such as a launch vehicle. It says what interface someone else must meet to use the launch vehicle. Can be anything, such as mass, type of connector, EMI…
  - Owned by manager of the system with which you want to interface
  - Probably not going to change
- Interface Requirement Document (IRD) - defines interfaces for two developing systems. Includes both physical and functional interfaces and ensures hardware & software compatibility.
  - Jointly managed (NEEDS ONE OWNER) and signed by the managers of the two systems in development.
- Interface Control Document (ICD) - Identifies the design solution for the physical interface (drawings).



Module Summary: Requirements — The Basics

- Requirements define the problem to be solved and establish the terms by which mission success will be measured.
- Requirements problems are the single biggest problem on development projects so care in creating good requirements always pays off.
- The later a problem is discovered the more costly it is to recover from.
- Requirements are distributed within the system architecture via flow-down, allocation and derivation.
- Requirements traceability is a technique of tracking the source and connections between requirements. It is used to assess the consequences of potential requirements changes.
- When a system is decomposed into smaller segments, interfaces are created that must be defined and managed.

Module Purpose: Writing Requirements

- Define and understand the characteristics of and rules for writing good requirements.
- Understand the value of providing the rationale and the preliminary verification technique with each requirement.
- Define and establish the difference between requirements verification and requirements validation.
- Establish the importance of resolving undefined (To Be Defined — TBD) and estimated (To Be Resolved — TBR) requirements early.

Related Readings:

Writing Good Requirements, INCOSE 1993, Ivy Hooks.

NASA Systems Engineering Handbook 2007, Appendix C, How to Write a Good Requirement

Good Requirements Are **SMART**

- **S**pecific -
  - It must address only one aspect of the system design or performance
  - It must be expressed in terms of the need (what and how well), not the solution (how).
- **M**easurable -
  - Performance is expressed objectively and quantitatively
  - E.g., an exact pointing requirement (in degrees) can be tested thus verified prior to launch.
- **A**chievable -
  - It must be technically achievable at costs considered affordable
  - E.g., JWST early designs specified an aperture requirement eventually descoped due to technical issues with deployment.
- **R**elevant -
  - It must be appropriate for the level being specified
  - E.g., requirement on the solar cells should not be designated at the spacecraft level.
- **T**raceable -
  - Lower level requirements (children) must clearly flow from and support higher level requirements (parents).
  - Requirements without a parent are referred to as orphans, and need to be assessed for necessity of inclusion.

Rules for Writing Good Requirements

- Requirements have mandatory characteristics:
  - *Needed*
  - *Verifiable*
  - *Attainable: technically, cost, schedule*
- Each requirement should
  - *Express one thought*
  - *Be concise and simple*
  - *Be stated positively*
  - *Be grammatically correct; free of typos and misspellings*
  - *Be understood only one way; they are unambiguous*
  - *Use consistent terminology to refer to the system/product and its lower level entities*
  - *Comply with the project's template and style rules*

More Rules for Writing Good Requirements

What a requirement must state:
- **WHO** is responsible
- **WHAT** shall be done
- Or **HOW WELL** something shall be done
- Or under what **CONSTRAINTS** something shall be done

Requirement format: **"WHO shall WHAT"**
- Uses active not passive voice

Example product requirements:
- The system shall operate at a power level of…
- The software shall acquire data from the…
- The structure shall withstand loads of…
- The hardware shall have a mass of…

Use the correct terms:
- *Requirements* are binding - **Shall**
- *Facts* or Declaration of purpose - **Will**
- *Goals* are non-mandatory provisions - **Should**

- Do NOT use "Must"

Goodness Checklist: Is this Requirement…

- Free of ambiguous terms?
  - Examples: as appropriate, etc., and/or, support, but not limited to, be able to, be capable of
- Free of indefinite pronouns?
  - Examples: this, these
- Free of unverifiable terms?
  - *Examples: flexible, user-friendly, robust, light-weight, maximize, adequate, small, portable, easily - other "ly" words and other "ize" words*
- *Free of implementation?*
  - *Requirement should state WHAT is needed, NOT HOW to provide it, i.e., state the problem not the solution.*
- *Necessary?*
  - *Ask "Why do you need the requirement?"; the answer may lead you to the real requirement.*
- *Free of descriptions of operations?*
  - *To distinguish between operations and requirements, ask "Does the developer have control over this?" "Is this a need the product must satisfy or an activity involving the product?"*
- *Free of TBDs (To Be Determined)?*
  - *Use a best estimate and a TBR (To Be Resolved) with rationale when possible.*

Example Requirements: Good or Bad?

- **The aircraft shall have three engines (initial DC-3 requirement).**
  - The aircraft shall meet the operation requirements with a single engine out.
- **The lunar lander shall include an airlock.**
  - The lunar lander shall provide the capability for crew to ingress/egress while maintaining pressurization.
- **The crew shall have the capability to perform extra-vehicular activities (EVAs).**
  - The vehicle shall allow extra-vehicular activities during operations.
- **The spacecraft shall maximize lifetime.**
  - The spacecraft shall have a lifetime of at least three years.
- **The software shall display data in a user-friendly fashion.**
  - The software shall display data as described in ICD 2345 Table 3.1.

Rationale Captures the Motivation and Assumptions of a Requirement

The rationale of each requirement defines

- Why a requirement is needed
- What assumptions were made
- What design effort drove the requirement
- Other data that will be needed to maintain the requirement over time

Example

- Requirement: "The truck shall have a height of no more than 14 feet."
- Rationale: 99% of all US interstate highway overpasses have a 14 foot or greater clearance. (Assumptions: The truck will be used primarily on US interstate highways for long-haul freight in the US.)

Space Example Requirement & Rationale

- Requirement: "The Constellation Architecture shall provide communication and tracking services prior to launch, during all mission phases, through recovery."
- Rationale: Communication and tracking services must be provided to elements during all mission phases. Ground based assets can provide this capability to elements when in direct line-of-sight with Earth. The Space Network can support launch, operations in LEO, re-entry and landing. Earth-based comm. and tracking services cannot, however, support the lunar landing, lunar ascent, or lunar surface operations if they are at polar locations or on the far side of the Moon. Another infrastructure element, such as a lunar relay, must be provided to support communications and tracking to elements during operations on the lunar far side or at the poles.
  - This requirement does not imply continuous coverage.
  - Lower level requirements will specify continuity and type of coverage.

Early Consideration of Requirement Verification Helps Avoid Costly Problems

- The systems engineer should create a preliminary verification plan as each requirement is written. Typically this plan is no more than establishing the technique (test, demonstration, analysis or inspection) that will be used for verifying a requirement.
- This early consideration of requirement verification helps:
  - Confirm the requirement is indeed verifiable
  - Define the system verification plan
  - Identify needed facilities for subsystem and system test

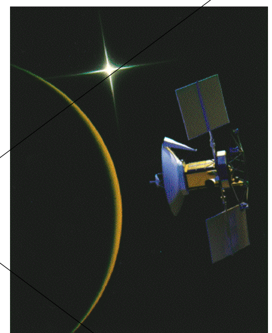
Preliminary Verification Plans Help Eliminate Impossible or Impractical Situations

The Magellan synthetic aperture radar had dozens of command variables each with hundreds of possible values and 8 redundant, cross-strapped subsystems. The number of hardware and command options made it impossible to test every valid radar command with every configuration.

The solution was to test 4 different hardware configurations with a small subset of stressing command variables at the extreme operating temperatures.

This helped establish confidence that the radar system would work with any valid command and hardware configuration while in orbit around Venus.

Magellan met or exceeded all of its mission requirements and mapped Venus from 1989 until its termination in 1994.



Templates Help Capture Verification Technique, Rationale, Significance and Traceability

Requirement Definition Card			
Requirement ID#:	*Unique alphanumeric*	Requirement Origin (Spec, Conop, etc.):	*Pointer to origin*
Requirement:	*Requirement statement*		
Source:	*Author*	Date:	*Origination Date*
Verification:	*How will the requirement be tested? One or two sentences.*		
Customer Satisfaction:	*(1-5) 5=essential to have, 1=indifferent to capability*	Customer Dissatisfaction:	*(1-5) 5=very dissatisfied if not implemented, 1=not concerned if not implemented*
Estimated Implementation Cost:	(1-5) 5=expensive to build, 1=inexpensive to implement	Estimated Implementation Risk:	*(1-5) 5= very high risk to implement, 1=no risk, 2= low risk*
Related Requirement(s):	*Identifier*	Conflicting Requirement(s):	*Identifier*
Rational:	*Pointer to info*	History:	*Origin, changes, or deletion (as applicable)*
Customer Approval:	*Name* *Date*	Provider Approval:	*Name* *Date*

Templates Help Capture Verification Technique, Rationale, Significance and Traceability

Requirement Definition Card			
Requirement ID#:	*Unique alphanumeric*	Requirement Origin (Spec, Conop, etc.):	*Pointer to origin*
Requirement:	*Requirement statement*		
Source:	*Author*	Date:	*Origination Date*
Verification:	*How will the requirement be tested? One or two sentences.*		
Customer Satisfaction:	*(1-5) 5=essential to have, 1=indifferent to capability*	Customer Dissatisfaction:	*(1-5) 5=very dissatisfied if not implemented, 1=not concerned if not implemented*
Estimated Implementation Cost:	(1-5) 5=expensive to build, 1=inexpensive to implement	Estimated Implementation Risk:	*(1-5) 5= very high risk to implement, 1=no risk, 2= low risk*
Related Requirement(s):	*Identifier*	Conflicting Requirement(s):	*Identifier*
Rational:	*Pointer to info*	History:	*Origin, changes, or deletion (as applicable)*
Customer Approval:	*Name* *Date*	Provider Approval:	*Name* *Date*

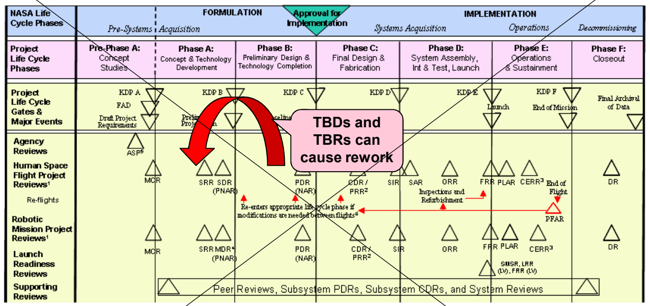
Templates Help Capture Verification Technique, Rationale, Significance and Traceability

Requirement Definition Card			
Requirement ID#:	*Unique alphanumeric*	Requirement Origin (Spec, Conop, etc.):	*Pointer to origin*
Requirement:	*Requirement statement*		
Source:	*Author*	Date:	*Origination Date*
Verification:	*How will the requirement be tested? One or two sentences.*		
Customer Satisfaction:	*(1-5) 5=essential to have, 1=indifferent to capability*	Customer Dissatisfaction:	*(1-5) 5=very dissatisfied if not implemented, 1=not concerned if not implemented*
Estimated Implementation Cost:	(1-5) 5=expensive to build, 1=inexpensive to implement	Estimated Implementation Risk:	*(1-5) 5= very high risk to implement, 1=no risk, 2= low risk*
Related Requirement(s):	*Identifier*	Conflicting Requirement(s):	*Identifier*
Rational:	*Pointer to info*	History:	*Origin, changes, or deletion (as applicable)*
Customer Approval:	*Name* *Date*	Provider Approval:	*Name* *Date*

Requirements Validation

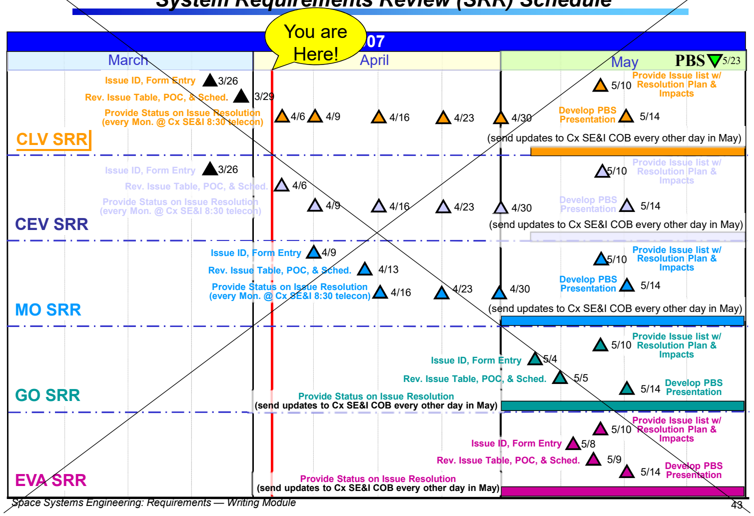
- Requirements validation asks three questions:
  - *Do we have the correct problem?*
  - *Do our requirements capture this problem?*
  - *Are our requirements SMART?*
- Performed to *assure* the system requirements *set* is complete, consistent, and each requirement is achievable and verifiable.
- Performed by subject matter experts, the system performing organization and the system authorized customer.
- Does the requirement set completely address and accomplish the mission?
- Are the requirements consistent with the established system boundaries and constraints?
- When does requirement validation take place?
  - *Usually before the System Requirements Review (SRR)*

Resolve TBDs and TBRs Early in the Project Life Cycle



- **TBDs and TBRs identify missing or uncertain information.**
- **When resolved they may cause rework of items previously completed.**
- **Assign responsibility for all TBDs and TBRs.**
- **Resolve as soon as possible.**

Overall Constellation Program System Requirements Review (SRR) Schedule



Constellation Program Status (spring '07) (Closure of SRR)

Board AI
- Total AI Count: 48 Total
- 9 Closed
- 39 Open
  - 30 are "Past Due"
  - 9 are in work, "Not Due Yet"

RIDS
- Total RID Count: 6,283
- 6,225 Closed
- 58 Open
  - Majority closed prior to PBS
  - All RIDs should be closed prior to CxP SDR

TBD/TBR
- Total Count: 2,532
  - Received plans for 2,064 TBRs/TBDs in 72 of 95 documents (76% of docs.)
  - Need burn-down plans for 196 TBRs/TBDs in 16 documents (17% of docs.)
  - 7 documents (274 TBRs/TBDs) will not be updated/baselined until post CxP PBS. These TBRs/TBDs will not be closed prior to CxP PBS

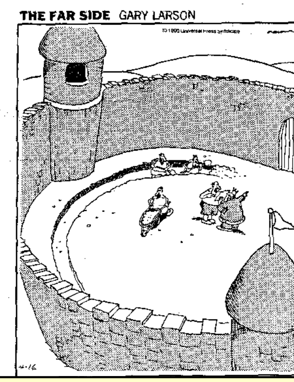
Document	Document Title	Being Worked
CxP 70008	Master Integration and Verification Plan (MIVP)	4
CxP 70023	CxP Prg Design Spec Natural Environments (DSNE)	9
CxP 70024	CxP HSIR	1
CxP 70036	Cx Environ Qual & Accept Testing Reqs (CEQATR)	5
CxP 70050 Vol 2	Electrical Power System Spec Vol 2	1
CxP 70061	C3I Strategic Plan	1
CxP 70067 Vol 1	CxP Program Human-rating Plan Vol 1	19
CxP 70080	Electromagnetic Environmental Effects (E3) Req'ts	6
CxP 70086	Software Verification and Validiation Plan	5
CxP 70135	Structural Design & Verification Requirements	7
Total		**58**

Work With the Customer to Distinguish Between ‘Desirements’ and Requirements

- *Desirement* – something that would be nice to have but is not mandatory to accomplish the mission
- Requirement - something that must be done for the mission to be successful
- Customers and stakeholders often blur the distinction between what is necessary and what would be nice to have

- Develop the problem statement independently of the solution
- Customers, stakeholders, design engineers and even systems engineers often have solution biases – Desirement space
- *Be Careful* - Customers or stakeholders may have a good idea, but early implementation solutions may prove impossible, impractical or sub-optimal when looked at in detail - capture what is necessary for mission success and *then* develop solutions to meet those needs.

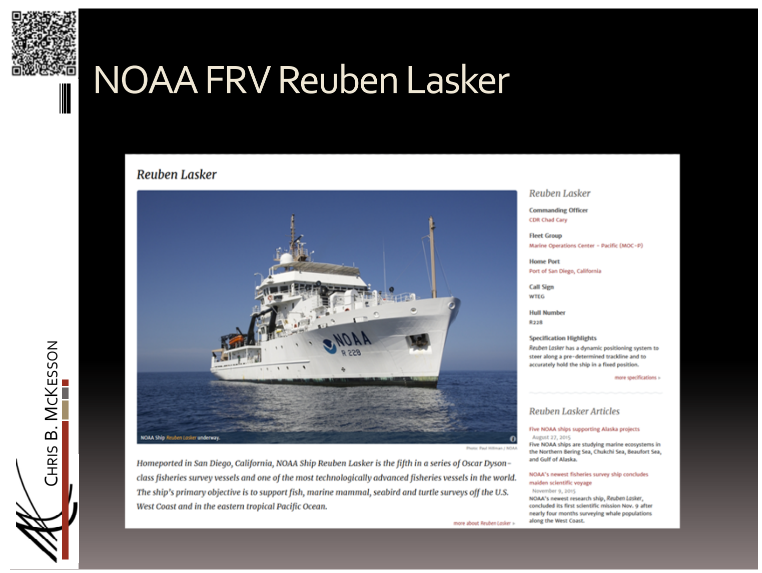
User Participation and Communication are Key



Suddenly, a heated exchange took place between the King and the moat contractor.

Module Summary: Writing Requirements

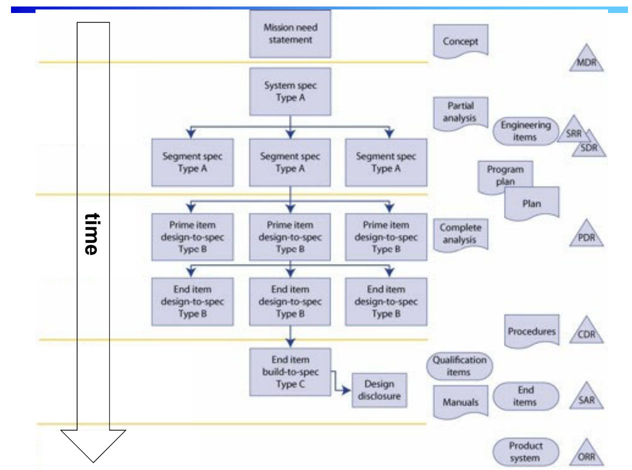
- Good requirements Are SMART: **S**pecific, **M**easurable, **A**chievable, **R**elevant and **T**raceable
- It is good practice to capture the rationale and preliminary technique for verification when writing requirements.
- Requirement validation is a process of ensuring that: the *set* of requirements is correct, complete, and consistent.
- Since the cost of reconciling undefined requirements grows as the project matures, undefined (TBD) and estimated (TBR) requirements should be resolved as early as possible.



Requirements Grow!

Chris B. McKesson

Evolution of Requirements



Space Systems Engineering: Requirements — Configuration & CM Module 50

Requirements Grow!

Growth must be managed!

Chris B. McKesson

Module Purpose: Configuration and Change Management

- *Define system baselines and when they are updated.*
- *Describe why system baselines are useful.*
- *Define requirements and configuration management and why they are necessary.*
- *Discuss the fact that changes are inevitable.*
- *Describe a typical management process for considering and assessing the impact of requirements and configuration changes.*

Baselines Periodically Capture the Complete System Representation

A system baseline is a description of the complete system including:

- requirements
- designs
- constraints
- assumptions
- interfaces
- resource allocations
- team responsibilities

at the time the baseline is created.

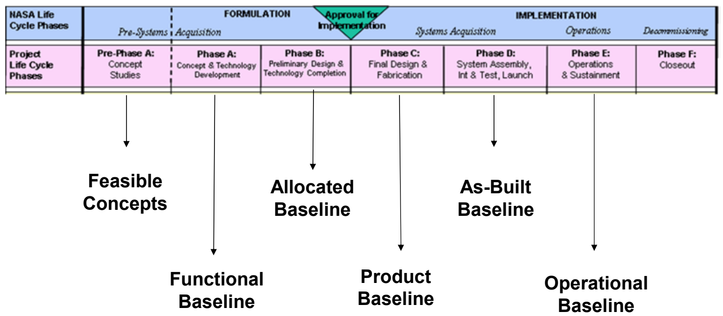
Baselines Help Ensure Everyone is on the Same Page

- With large teams working on many different parts of a project simultaneously, it is important to make sure there is a common understanding of what is to be done and that no necessary task is ignored.
- Baselines are established at milestone reviews (SDR, PDR, CDR, ORR) and are the common departure point for subsequent design and product maturation.
- Baselines also ensure that the entire project matures at an approximately uniform rate.
  - If one subsystem design is advanced much beyond its peers and it is later discovered that the allocations or interfaces are inappropriate, more rework will have to be done than if the subsystems had advanced at the same rate.

Baselines Help Ensure Everyone is on the Same Page

- With large teams working on many different parts of a project simultaneously, it is important to make sure there is a common understanding of what is to be done and that no necessary task is ignored.
- Baselines are established at milestone reviews (SDR, PDR, CDR, ORR) and are the common departure point for subsequent design and product maturation.
- Baselines also ensure that the entire project matures at an approximately uniform rate.
  - If one subsystem design is advanced much beyond its peers and it is later discovered that the allocations or interfaces are inappropriate, more rework will have to be done than if the subsystems had advanced at the same rate.

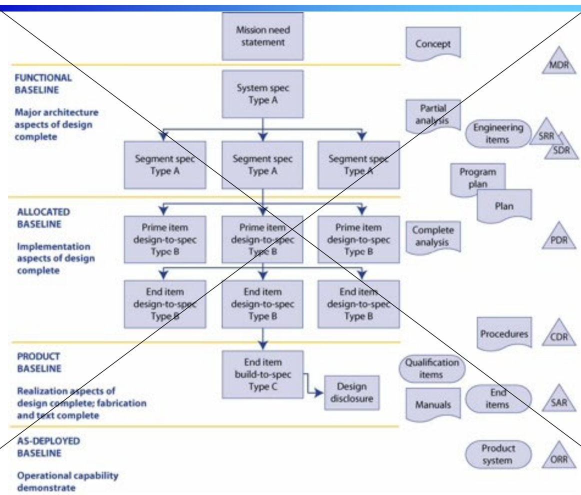
System Maturity Advances Over the Project Life Cycle



Technical Baseline Definitions

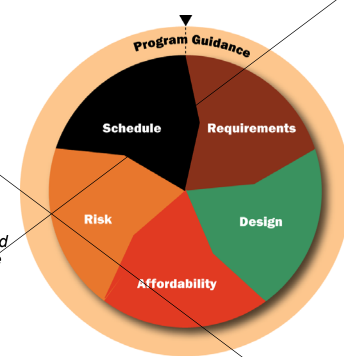
- Functional Baseline (Phase A)
  - The functional baseline is the approved documentation describing a system's functional, performance, and interface requirements and the verifications required to demonstrate achievement of those specified characteristics.
  - Established at the System Definition Review (SDR).
- Allocated Baseline aka the 'Design-to' Baseline (Phase B)
  - The allocated baseline extends the top-level performance requirements of the functional baseline to sufficient detail for initiating manufacturing or coding.
  - Established at the Preliminary Design Review (PDR).
- Product Baseline aka the 'Build-to' Baseline (Phase C)
  - The product baseline describes detailed form, fit, and function characteristics; the selected functional characteristics designated for production acceptance testing; the production acceptance test requirements.
  - Established at the Critical Design Review (CDR).
- 'As-Built' Baseline (Phase D)
  - The as-built baseline describes the detailed form, fit, and function of the system as it was built.
  - Established at the Flight Readiness Review (FRR).
- Operational Baseline aka 'As-Deployed' Baseline (Phase E)
  - The as-deployed baseline occurs at the Operational Readiness Review (ORR). At this point, the design is considered to be functional and ready for flight. All changes will have been incorporated into the final documentation.

Evolution of the Technical Baseline



Baselines Are More Than Just Requirements and Designs

- Technical baseline deals with requirements and design.
- New focus: Integrated program management synchronizes these baselines:
  - Requirements
  - Design
  - Affordability ($$$)
  - Schedule
  - Risk
- *All 5 baselines need to be linked and tracked over the project life cycle.*
- Use of tools and processes to ensure that the linkages and their impacts are captured and updated in all major project documents.
- This practice enables informed decision making for the future.

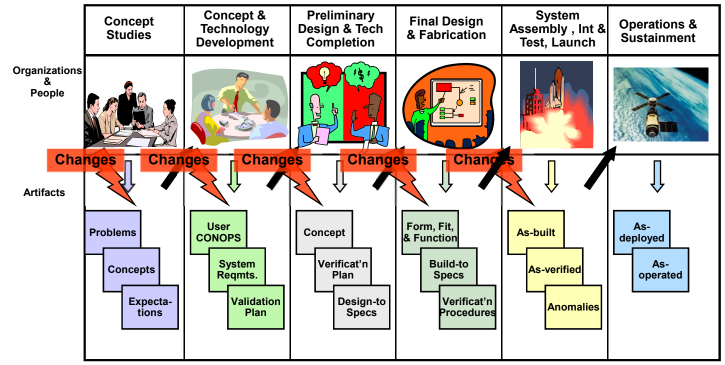


Integrated Program Management

Managing Requirements and the System Configuration is a Necessity

- Capturing all the requirements and their traceability can be a *mess*!
  - Parent requirements beget child requirements
  - Problem-space requirements beget solution-space requirements
  - Functional and performance requirements have lots and lots of peers
  - Traceability, linkages and rationale must be documented and maintained
  - So baselined requirements are required for each control gate
- Management of it all.
  - Configuration management keeps track of all of the requirements, and once hardware is built or software coded, keeps track of what has been built and coded.
  - This is a huge, complex and extremely important bookkeeping job – made easier today by database tools (e.g.,CRADLE or DOORS).

Change - The One Constant



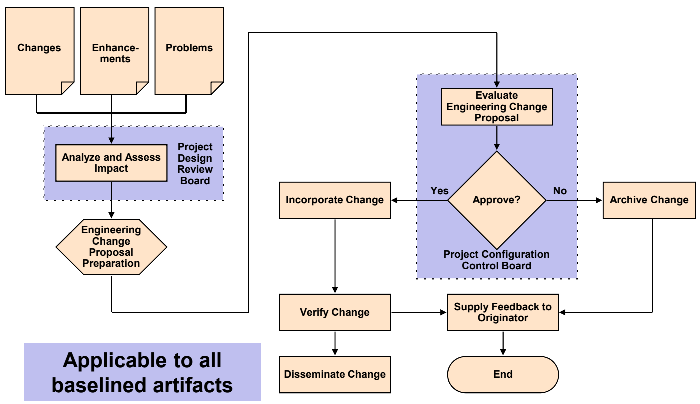
Where Do Changes Come From?

- Requirements change when:
  - They are reallocated as the system design matures, since initial allocations are typically suboptimal.
  - New requirements are added to the system, since initial requirements may not have been complete.
  - A stakeholder decides that new functions or performance is needed.
  - Measured performance does not meet requirements. Reallocation or redesign are possible responses to non-compliance in test.
- Configurations change when:
  - What is built is not identical to what is designed. Configuration descriptions strive to be the most accurate possible description of the current system.
  - Something breaks in test. Reallocation or redesign are possible responses to test failures.

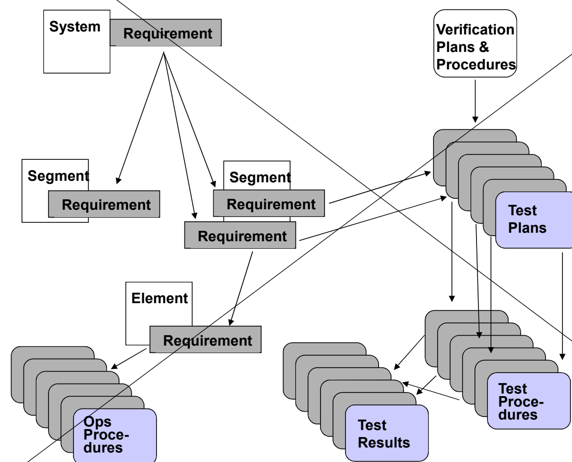
Requirements Change Control

- Capturing the complete set of requirements and assessing the impacts of considered changes are systems engineering responsibilities.
- Top level requirements are captured first, then lower levels as the system design matures.
  - *Top level requirements are typically placed under change control just after the System Requirements Review (SRR).*
  - *Lower level requirements are placed under change control after the corresponding subsystem Preliminary Design Review (PDR).*
- Engineering Change Requests (ECRs) are the means for making changes to requirements, with assessment and review.
- Change Control Boards (CCBs) are established to review and assess the impacts of ECRs.

Typical Requirement and Configuration Control Flowchart



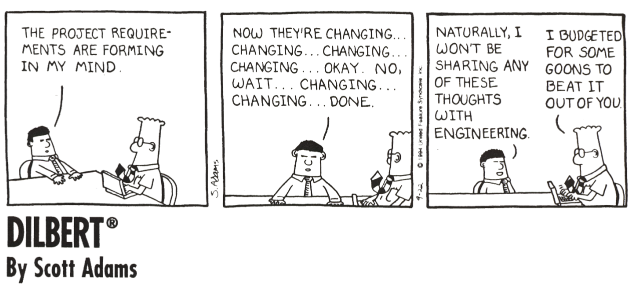
The Later a Change is Proposed the More Costly it is to Implement



- For every hour spent in writing requirements - hundreds of hours will be spent in reading requirements - plus
  - Endless meetings
  - Countless debates
  - Continual rewrites
  - Continual rework

=> $$$$$$

Mission Analysis, Like We Usually Have to Do It



Module Summary: Configuration and Change Management

- System baselines capture the complete, current system description.
- System baselines are updated periodically at five major milestone reviews - SDR, PDR, CDR, FRR and ORR.
- Requirements and configuration changes are inevitable, so a formal process of considering the implications of these changes is used.
- It is important to have managed baselines, requirements and configurations so that the entire team is working with the same assumptions of what the current system is and what it must do.
- Systems engineering is responsible for creating and updating the system baseline, assessing the implications of considered changes and disseminating the news of any accepted changes.

Backup Slides
for Requirements — Configuration &CM Module

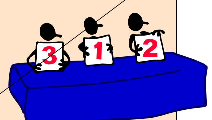
Prioritization

- List items that are mandatory.
- Group them as "musts."
- All other items are "wants" that can be prioritized.
- Important "wants" are given a high weighted value.
- When candidate concepts are evaluated, if they do not satisfy all the "musts," they are eliminated.

Be careful about overstating the "musts." Otherwise, promising candidates may be prematurely eliminated.

Prioritizing Wants

♦ Several methods:
- High, Medium, Low
  - Select highs and lows and all else falls into medium
- One, Two, Three
  - Same as high, medium, and low
- Relative to a base of ten
  - Relative importance assigned a number against a scale (0-10), with ten being the highest.
- Pair-wise comparison
  - Each "want" is compared to each other and a decision is made as to which one is more important. When all comparisons have been made a priority stacking results.
- Categorize "satisfaction" and "dissatisfaction"
  - "How pleased will you be when this capability is provided?"
  - "How upset will you be if we cannot provide this capability?"



Get the users involved to establish and baseline the priorities.

So What?

- Freeze your Requirements
- Freeze your Configuration
- Do some Engineering
- Come back together and see if the Requirements, or the Configuration, need to change

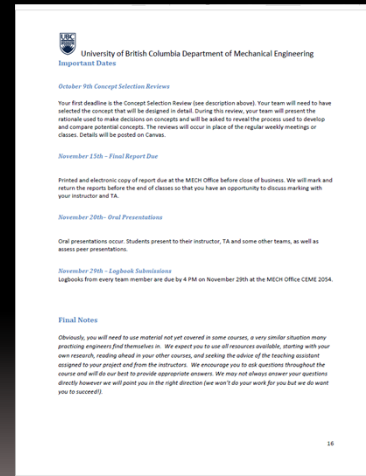
- Don't do it ad hoc!
- Do not change configuration "on the fly"

CHRIS B. MCKESSON

When shall we do it?

- Let's look…

CHRIS B. MCKESSON



University of British Columbia Department of Mechanical Engineering

Important Dates

October 9th Concept Selection Reviews

Your first deadline is the Concept Selection Review (see description above). Your team will need to have selected the concept that will be designed in detail. During this review, your team will present the rationale used to make decisions on concepts and will be asked to reveal the process used to develop and compare potential concepts. The reviews will occur in place of the regular weekly meetings or classes. Details will be posted on Canvas.

November 15th – Final Report Due

Printed and electronic copy of report due at the MECH Office before close of business. We will mark and return the reports before the end of classes so that you have an opportunity to discuss marking with your instructor and TA.

November 20th– Oral Presentations

Oral presentations occur. Students present to their instructor, TA and some other teams, as well as assess peer presentations.

November 29th – Logbook Submissions

Logbooks from every team member are due by 4 PM on November 29th at the MECH Office CEME 2054.

Final Notes

Obviously, you will need to use material not yet covered in some courses, a very similar situation many practicing engineers find themselves in. We expect you to use all resources available, starting with your own research, reading ahead in your other courses, and seeking the advice of the teaching assistant assigned to your project and from the instructors. We encourage you to ask questions throughout the course and will do our best to provide appropriate answers. We may not always answer your questions directly however we will point you in the right direction (we won't do your work for you but we do want you to succeed!).

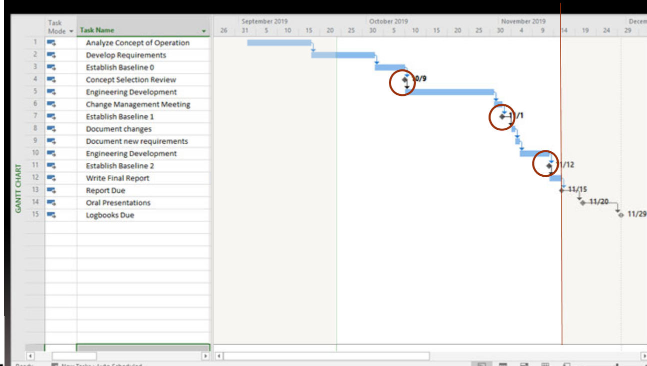
16

CHRIS B. MCKESSON

I recommend:

- Freeze Baseline 0 at CSR
  - Develop Design 1
  - Solve known unknowns
  - Document unk/unks discovered
  - Adjudicate Baseline 1
- Freeze Baseline 1 o/a November 1
  - Correct deficiencies / changes
  - Document Baseline 2
  - Finalize Report
- Final report due 15 November

CHRIS B. MCKESSON



CHRIS B. MCKESSON

Chris B. McKesson