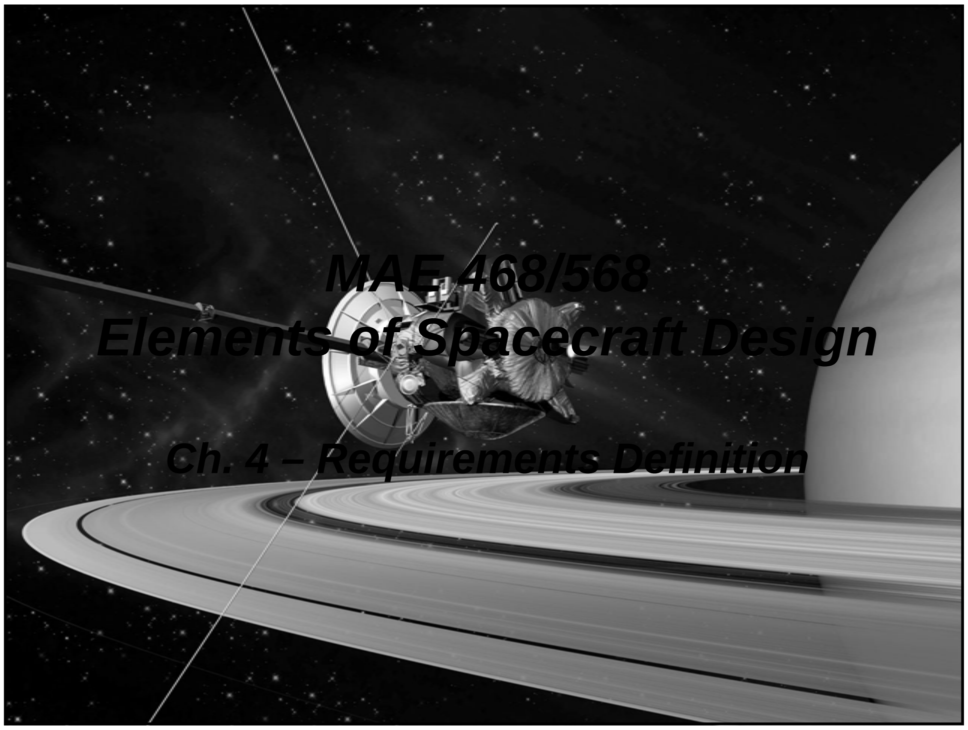


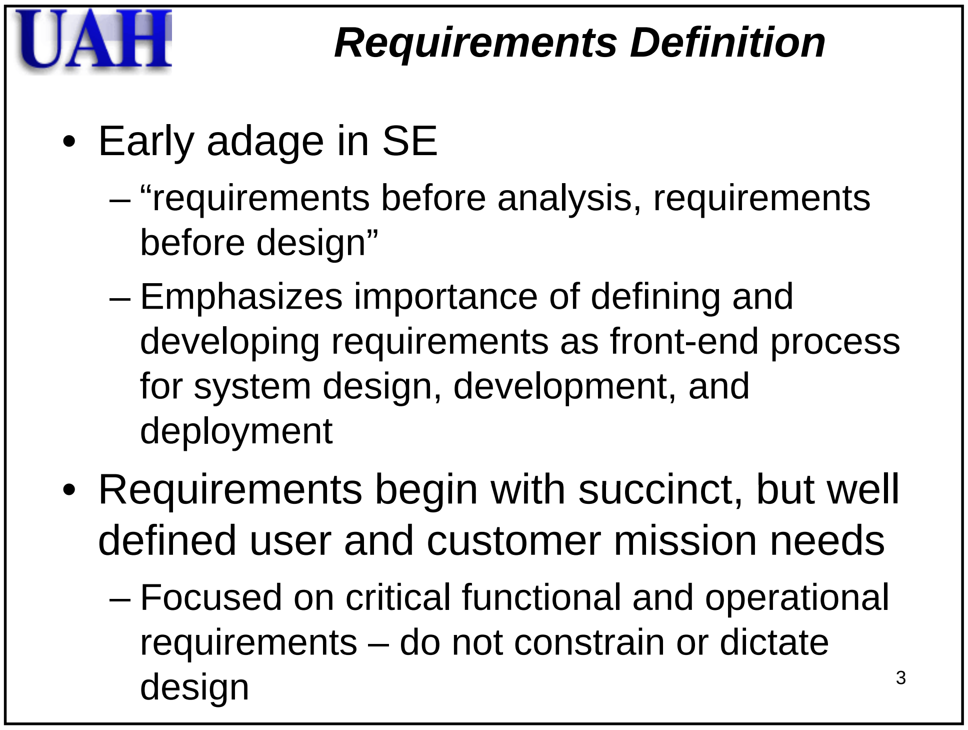
MAE 468/568 Elements of Spacecraft Design

Ch. 4 – Requirements Definition

Class Agenda

- Role of requirements in system development
  - QFD – a tool of requirements development
- Requirements analysis and performance budgeting
  - Functional analysis
  - Initial performance budgets
  - Refining and negotiating performance budgets
- Requirements documentation and specifications
- Steps to a requirements baseline

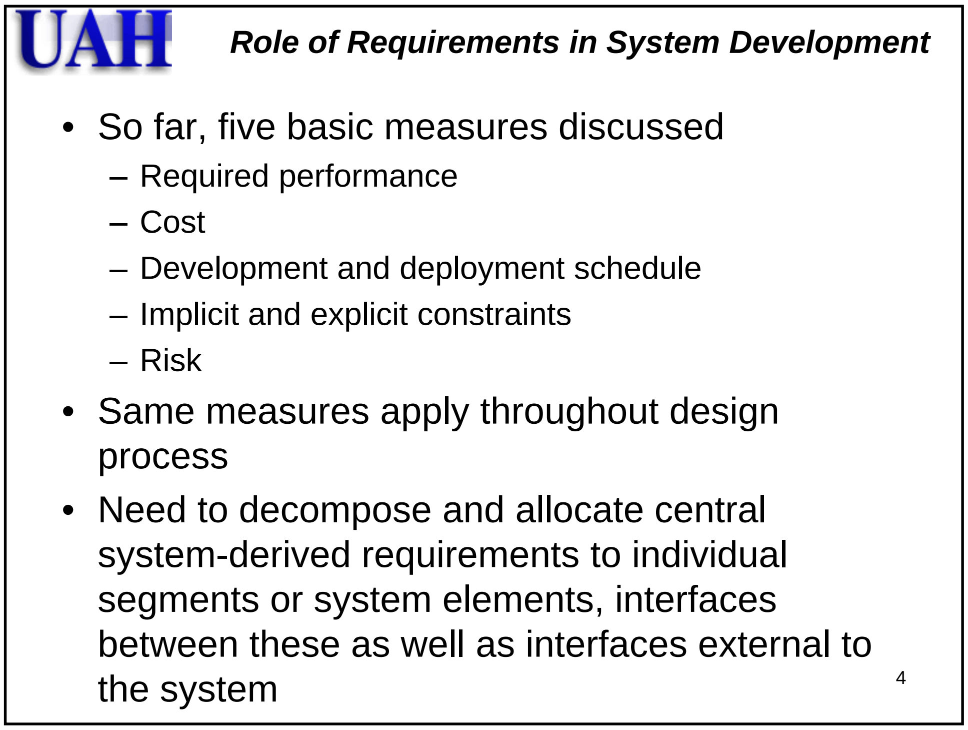
2



Requirements Definition

- Early adage in SE
  - “requirements before analysis, requirements before design”
  - Emphasizes importance of defining and developing requirements as front-end process for system design, development, and deployment
- Requirements begin with succinct, but well defined user and customer mission needs
  - Focused on critical functional and operational requirements – do not constrain or dictate design

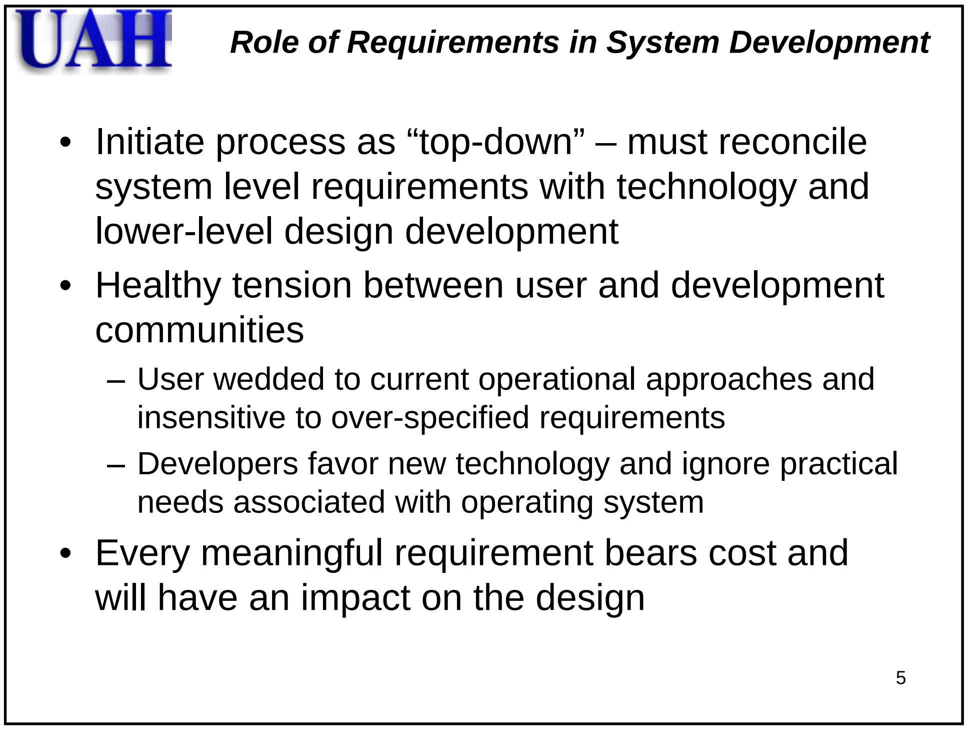
3



Role of Requirements in System Development

- So far, five basic measures discussed
  - Required performance
  - Cost
  - Development and deployment schedule
  - Implicit and explicit constraints
  - Risk
- Same measures apply throughout design process
- Need to decompose and allocate central system-derived requirements to individual segments or system elements, interfaces between these as well as interfaces external to the system

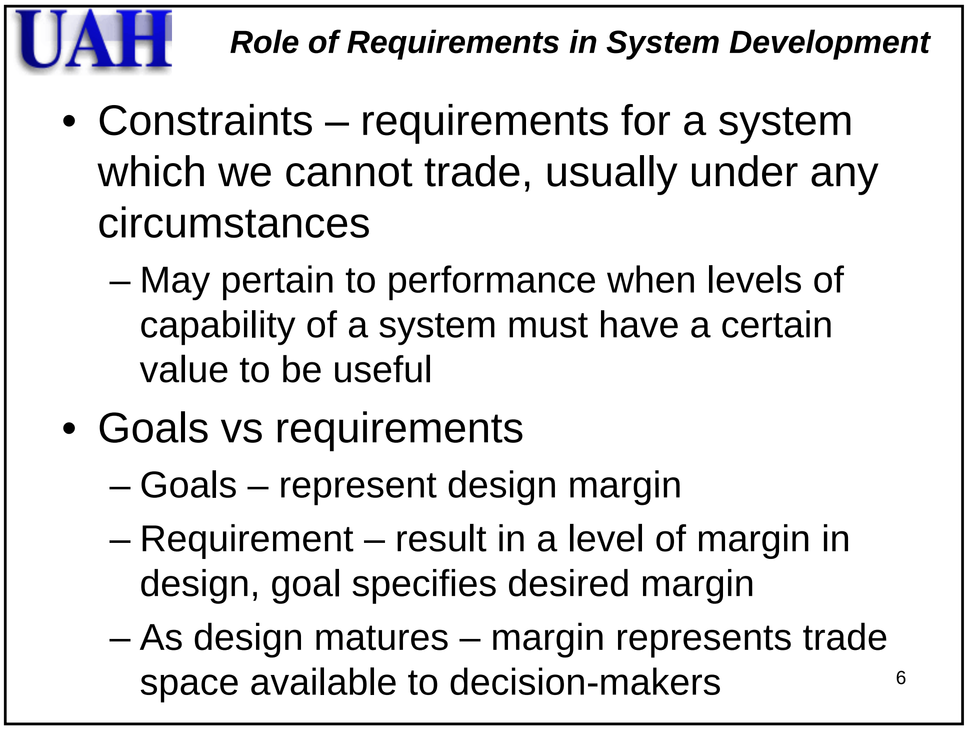
4



Role of Requirements in System Development

- Initiate process as “top-down” – must reconcile system level requirements with technology and lower-level design development
- Healthy tension between user and development communities
  - User wedded to current operational approaches and insensitive to over-specified requirements
  - Developers favor new technology and ignore practical needs associated with operating system
- Every meaningful requirement bears cost and will have an impact on the design

5



Role of Requirements in System Development

- Constraints – requirements for a system which we cannot trade, usually under any circumstances
  - May pertain to performance when levels of capability of a system must have a certain value to be useful
- Goals vs requirements
  - Goals – represent design margin
  - Requirement – result in a level of margin in design, goal specifies desired margin
  - As design matures – margin represents trade space available to decision-makers

6

Role of Requirements in System Development

- Designers focus on performance areas
  - – Less focus on mundane requirements (availability, accommodation to external environment)
- Role and characteristics of requirements change in each development phase

7

Evolution of Requirements

- Needs analysis
  - – Defining mission requirements
  - – Defining environment
  - – Identifying mission drivers and constraints
  - – Technology programs
- Concept development
  - – Identifying critical driving requirements and associated risks
  - – Developing operations and design concepts
  - – Cost estimates
  - – Functional analysis and major interfaces
  - – System studies and simulations
  - – Prototyping and assessing technology

8

Evolution of Requirements

- Concept validation
  - Tailored system and segment definitions
  - Preliminary internal interface requirements
  - Preliminary system standards
  - Preliminary requirements flowdown
  - Integrated system validation including test planning
  - Transition planning
  - Validating technology
- Design and implementation
  - Detailed requirements flowdown
  - Developing formal design documentation and interface control
  - Integrating and testing the system
  - Demonstrating and verifying the system
  - Test procedures and reports

9

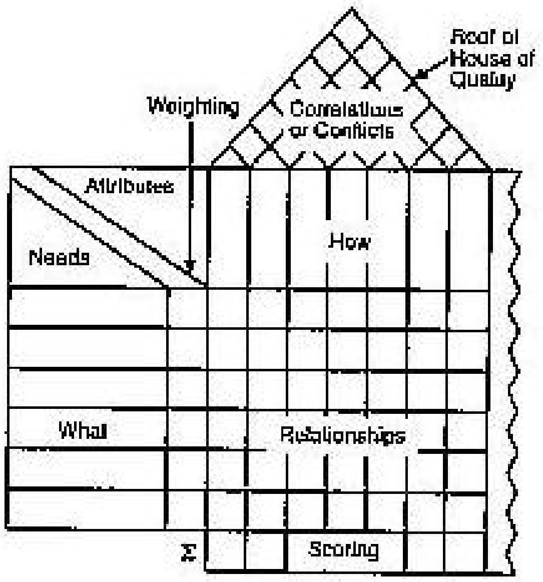
QFD – A Tool for Requirements Development

- QFD
  - Quality or features
  - Function or mechanization
  - Deployment or evaluation
- "attribute and function development"
- Series of matrices organized to define system characteristics and attributes
- Applicable over multiple areas of decomposition
- *House of Quality* – 1st level connecting customer needs/requirements to technical attributes/requirements

10

UAH

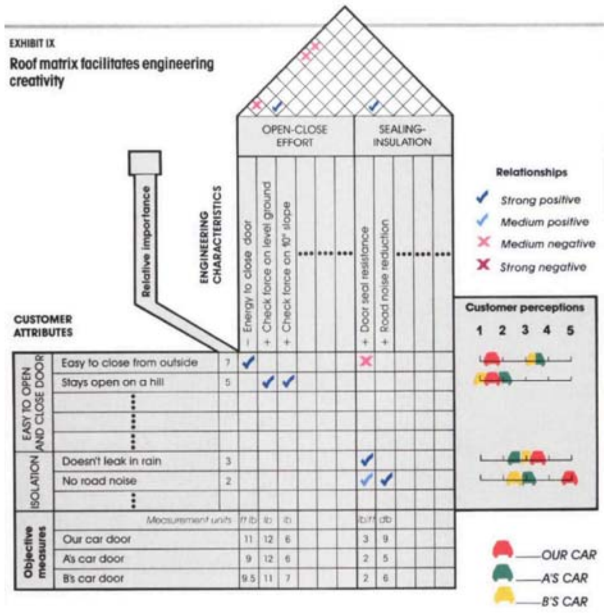
QFD – A Tool for Requirements Development



11

UAH

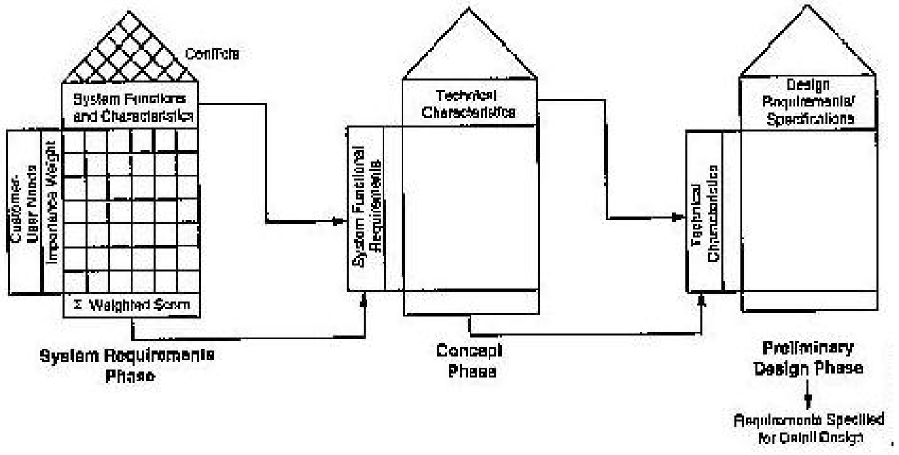
QFD – A Tool for Requirements Development



12

UAH

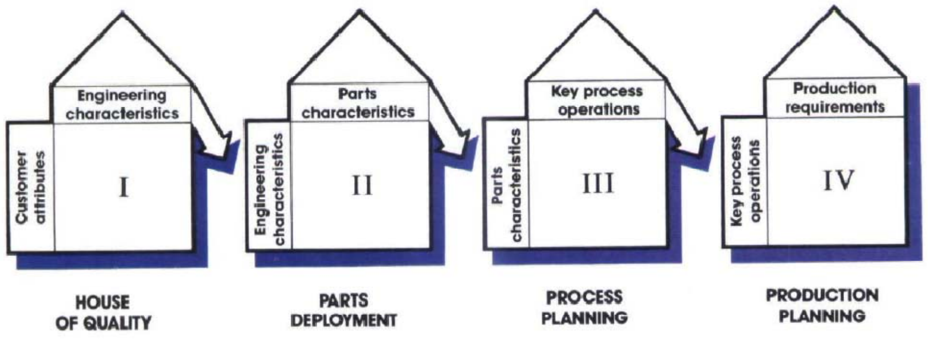
Progression of QFD


Conflicts
System Functions and Characteristics
Customer-User Needs
Importance Weight
Σ Weighted Score
System Requirements Phase
Technical Characteristics
System Functional Requirements
Concept Phase
Design Requirements/ Specifications
Technical Characteristics
Preliminary Design Phase
Requirements Specified for Detail Design

13

UAH

Progression of QFD



14

QFD – A Tool for Requirements Development

- QFD
  - Structured means for a design team to address customer needs and to develop consequent design characteristics to satisfy them
  - Serves to sustain trail of requirements derivation
  - Provides means for analyze impact of changes to requirements at any level
  - Logical translation to functional analysis via functional flow diagrams

15

Requirements Analysis & Performance Budgeting

- Must decompose every system requirement into progressively lower levels of design by defining lower level functions which determine how each function must perform
- Allocation – assigns function and its associated performance requirements to a lower level design element
- Starts @ system level – requirements derive directly from mission needs – proceeds through segment, subsystem, and component design levels

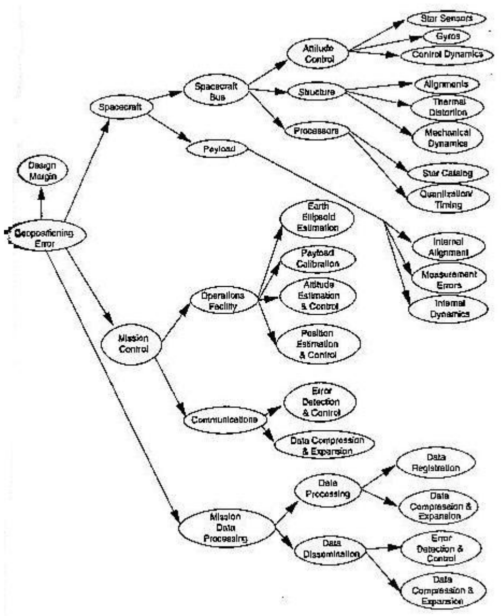
16

Requirements Analysis & Performance Budgeting

- Must ensure closure at next higher level
- Closure – satisfying lower-level requirements ensures performance at next higher level – can trace all requirements back to satisfying mission needs
- Emphasizes iterative character of requirements development process

17

Requirements Allocation



18

UAH

Functional Analysis

- Simplest way to represent functions – functional-flow block diagram (FFBD)
- Define topmost or first level functions of system in sequence in which they occur
- Successive decomposition permits ID how system works at each level before proceeding to lower levels
- Once top-level functions are established, can decompose/analyze each function throughout remaining layers of flow
- Functional decomposition – necessary in allocating design characteristics at each level of system architecture

19

UAH

Functional Analysis

- Permits allocation of performance budgets together with other budgets affecting cost and risk
- FFBDs can be used to depict information/data flow
  - Interface data flowing between functions, control relationships, data sources/destinations

20

Initial Performance Budgets

- Analyzing requirements leads to hierarchically organized performance metrics and budgets for interactive development segments
- Begins with budgets derived using analysis, simulation, known design or test data, large measure of experience
- Mission drivers should be primary drivers in development of requirements and derived functions
- Experience/related reference missions important in developing initial performance budgets to meet system performance requirements

21

Initial Performance Budgets

Primary	**Secondary**
Weight	Subsystem weight Power Propellant
Geolocation or System Pointing Errors	Pointing & Alignment Mapping Attitude control Attitude determination Position determination
Timing	Coverage Communications Operations Processing
Availability	Reliability Operations
Cost	Development cost Deployment cost Operations and maintenance cost

UAH

Initial Performance Budgets

- Budgeted items
  - Directly from requirements
  - Related to elements of overall system design
- Timeline budgets at system level are typical mission drivers
- Design budgets developed by systems engineers with broad understanding of system and its elements
- Details of new technology and lower level design studies can/should result in adjustments to budgets as experts review initial allocations

23

UAH

Initial Performance Budgets

- Key aspect of system design
  - Robust initial allocation, adaptable to iterations
- Important
  - Recognize iterative nature of process
  - System solution may impose stringent demands on certain aspects of lower level designs than others
- Process involves a high degree of negotation

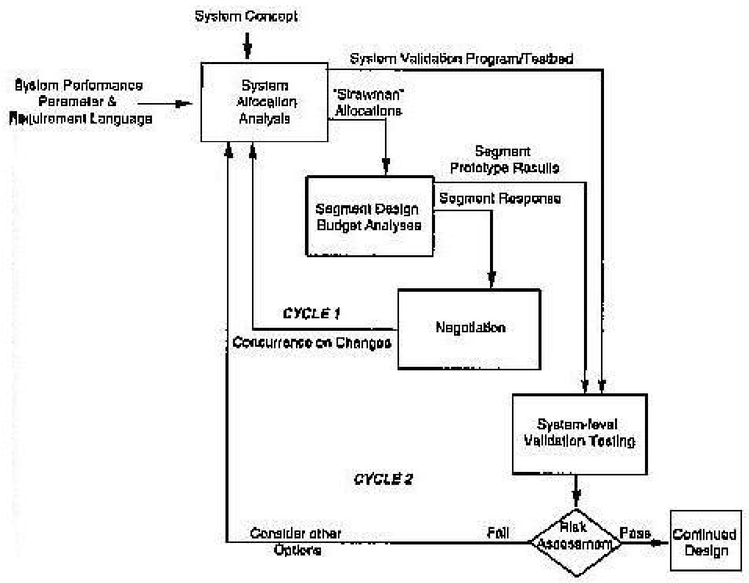
24

Refining and Negotiating Performance Budgets

- Systems engineers must thoroughly understand how to develop and define requirements, then allocate and negotiate budgets associated with them
- Failure to meet key budgets can lead to major system problems
- Early definition permits iterative process of adjusting allocations, margins, and even operations well before major cost/schedule penalties occur
- Performance budgeting and validating key system requirements is iterative process

25

Refining and Negotiating Performance Budgets


System Concept
System Validation Program/Testbed
System Performance Parameter & Requirement Language
System Allocation Analysis
"Strawman" Allocations
Segment Prototype Results
Segment Design Budget Analyses
Segment Response
CYCLE 1
Concurrence on Changes
Negotiation
System-level Validation Testing
CYCLE 2
Consider other Options
Fail
Risk Assessment
Pass
Continued Design

26

Refining and Negotiating Performance Budgets

- Before – specific performance parameter/requirement must be clear and traceable to mission need
- QFD – good tool to do this
- Vague, inconsistent, or unquantifiable requirements too often lead to inaccurate understanding, misinterpretation and/or exploitation
- Applies to critical areas of system performance – becomes expensive
- Iterative process includes negotiation and re-negotiation of budgets

27

Refining and Negotiating Performance Budgets

- System level design is logical integration/synthesis of segment design
- Defining functions, performance requirements, and interfaces requiring support lays framework for deciding “how” to design each segment
- System/segment specifications provide system/interface requirements
- Lower level specifications provide design requirements
- System engineer and segment designers must interact at all design levels

28

Refining and Negotiating Performance Budgets

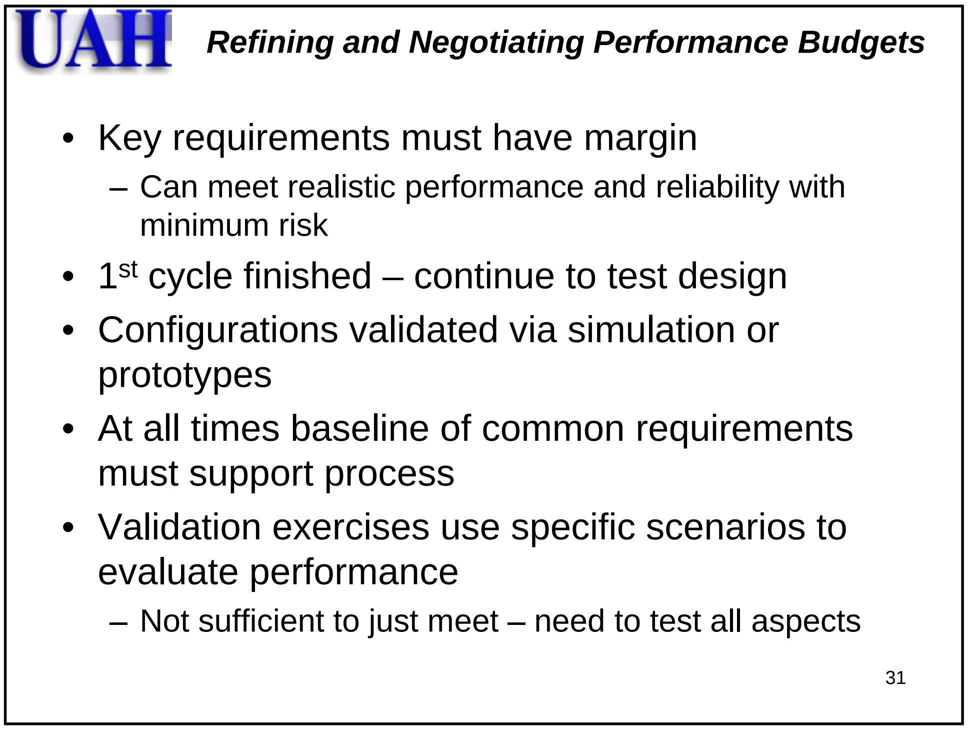
- Initial budget estimates almost never correspond with design considerations at lower levels
- Early budgets are starting points for negotiation and budget adjustment – to reconcile early system allocations with segment estimates of design and performance
- As reconciliation occurs – document in requirements reference which changes only with full traceability and visibility for all stakeholders
- Each critical performance parameter matches an established system and segment budget

29

Refining and Negotiating Performance Budgets

- Budgets can and normally do change as developers proceed on design and validate performance
- At this stage design margin becomes issue
  - How much to keep
  - Who know where it is
  - Who has authority to adjust it
- Typically margin is statistical (2σ)
  - Can produce significant overdesign and cost as it flows through design
  - Design engineers may keep margin at lower levels – not visible – can complicate design

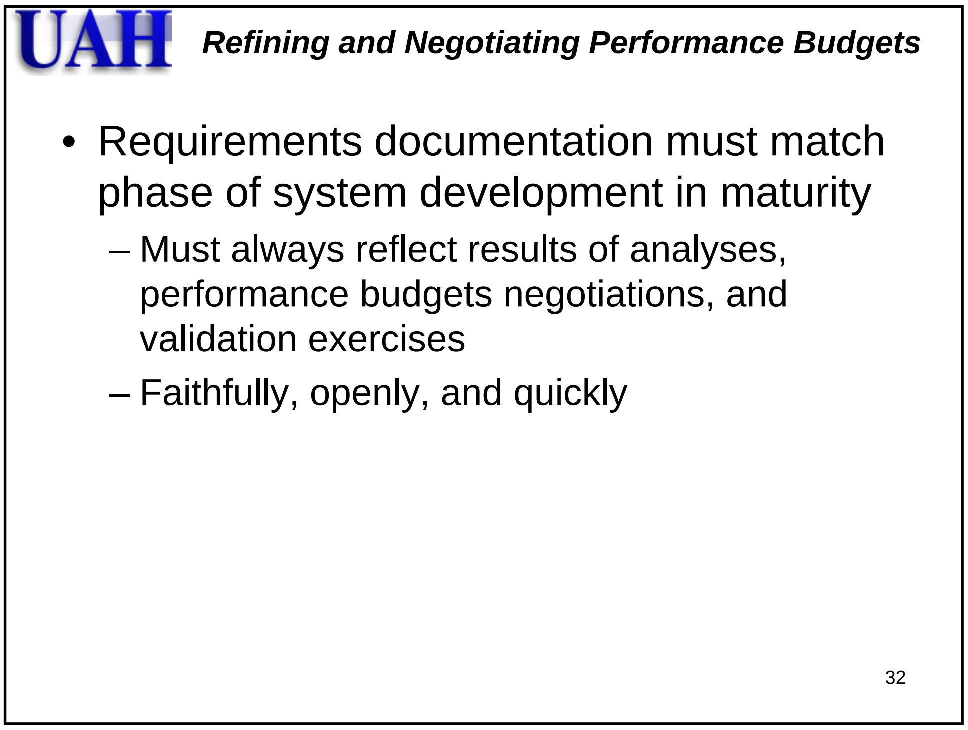
30



Refining and Negotiating Performance Budgets

- Key requirements must have margin
  - Can meet realistic performance and reliability with minimum risk
- 1st cycle finished – continue to test design
- Configurations validated via simulation or prototypes
- At all times baseline of common requirements must support process
- Validation exercises use specific scenarios to evaluate performance
  - Not sufficient to just meet – need to test all aspects

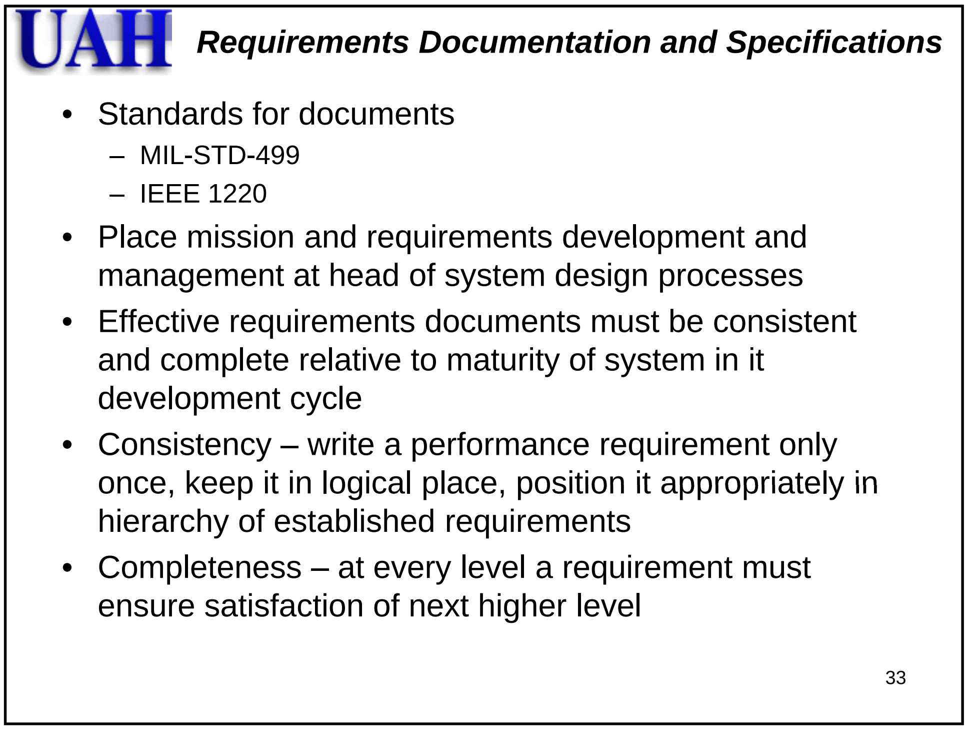
31



Refining and Negotiating Performance Budgets

- Requirements documentation must match phase of system development in maturity
  - Must always reflect results of analyses, performance budgets negotiations, and validation exercises
  - Faithfully, openly, and quickly

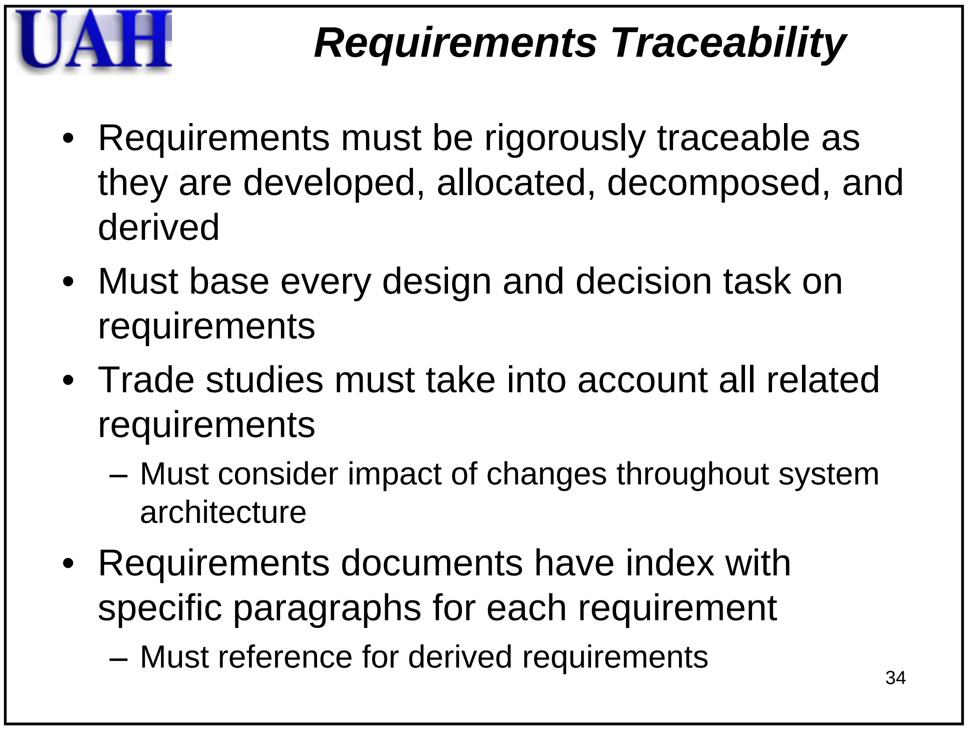
32



Requirements Documentation and Specifications

- Standards for documents
  - MIL-STD-499
  - IEEE 1220
- Place mission and requirements development and management at head of system design processes
- Effective requirements documents must be consistent and complete relative to maturity of system in it development cycle
- Consistency – write a performance requirement only once, keep it in logical place, position it appropriately in hierarchy of established requirements
- Completeness – at every level a requirement must ensure satisfaction of next higher level

33



Requirements Traceability

- Requirements must be rigorously traceable as they are developed, allocated, decomposed, and derived
- Must base every design and decision task on requirements
- Trade studies must take into account all related requirements
  - Must consider impact of changes throughout system architecture
- Requirements documents have index with specific paragraphs for each requirement
  - Must reference for derived requirements

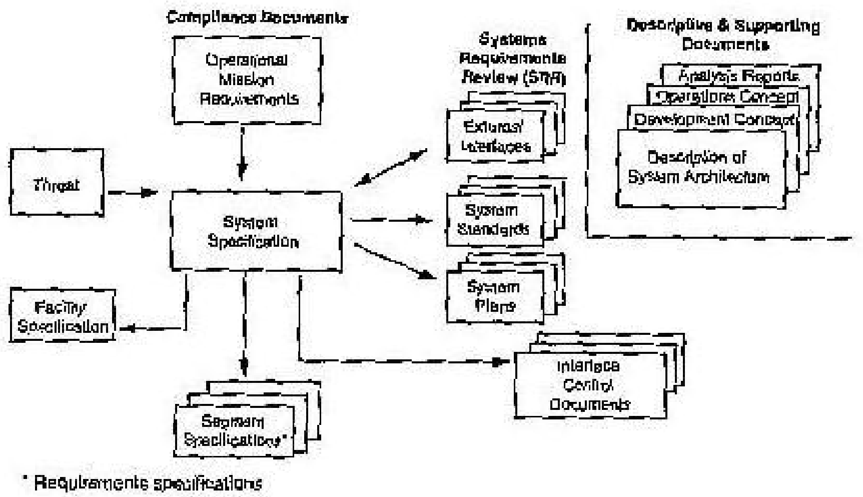
34

Requirements Traceability

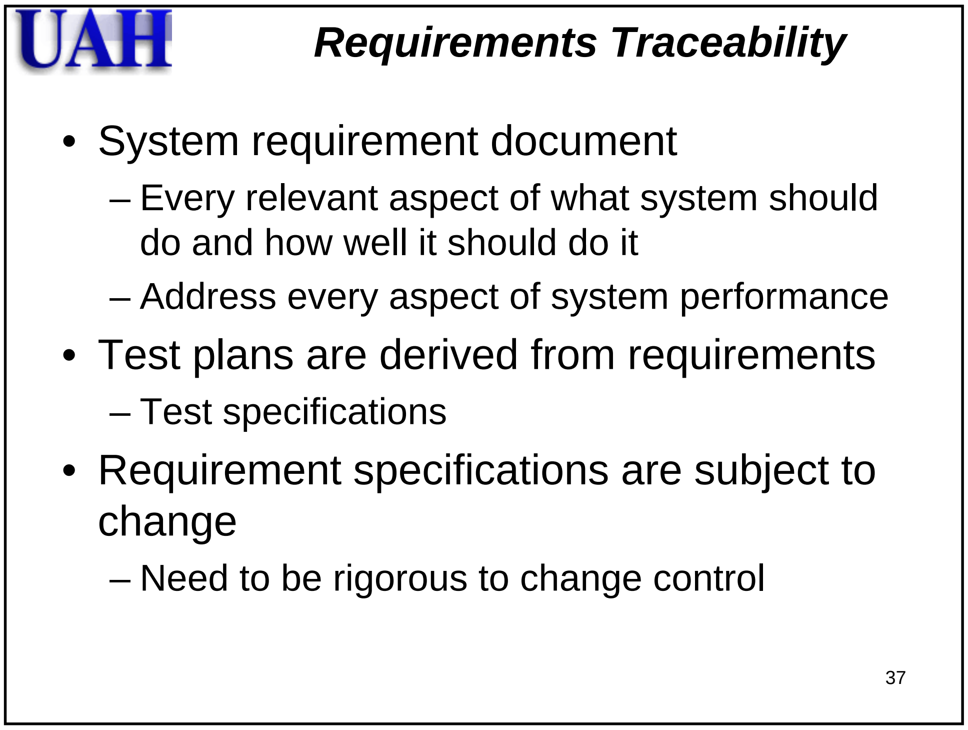
- Traceability emphasizes need for effective requirements to be unambiguous and verifiable to avoid misinterpretation and exploitation
  - “optimize” or “minimize” should not be requirements
- Will have requirements reviews just like design reviews
- Requirements documentation falls into nine classes (see next chart)

35

Requirements Traceability



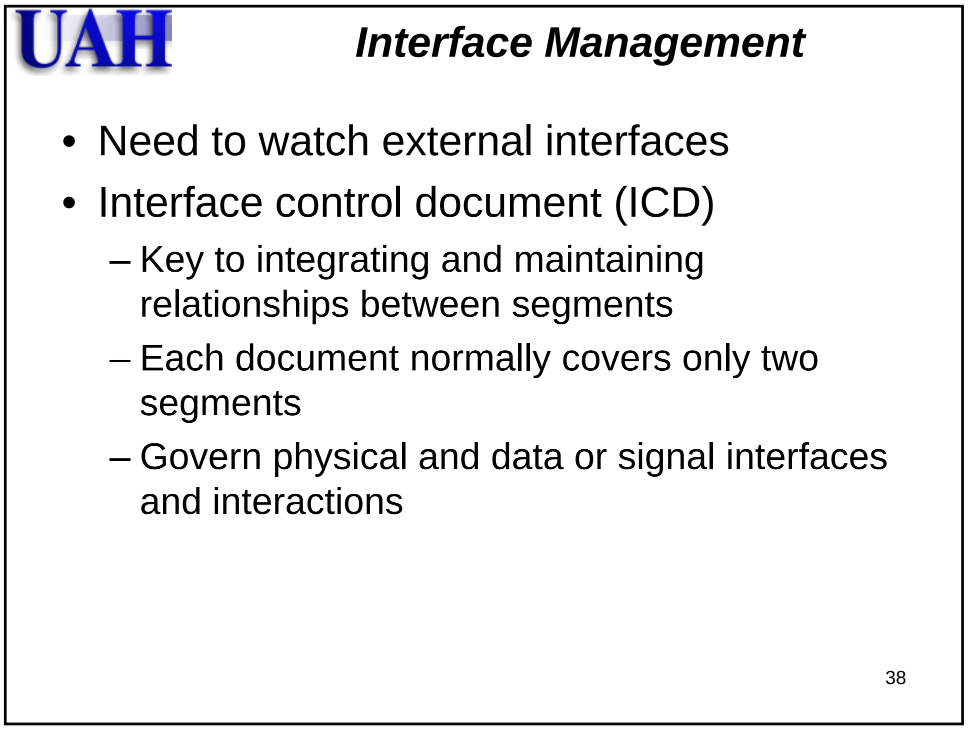
36



Requirements Traceability

- System requirement document
  - Every relevant aspect of what system should do and how well it should do it
  - Address every aspect of system performance
- Test plans are derived from requirements
  - Test specifications
- Requirement specifications are subject to change
  - Need to be rigorous to change control

37



Interface Management

- Need to watch external interfaces
- Interface control document (ICD)
  - Key to integrating and maintaining relationships between segments
  - Each document normally covers only two segments
  - Govern physical and data or signal interfaces and interactions

38

UAH *Steps to Develop a Requirements Baseline*

1. Identify customer and user of product
2. Identify and prioritize customer/user objectives and needs for mission to be accomplished
3. Define internal and external constraints
4. Translate customer/user needs into functional attributes and system characteristics
5. Establish functional requirements for system and provide for decomposition to elements
6. Establish functional flow and requirements for its performance of functions

39

UAH *Steps to Develop a Requirements Baseline*

7. Translate functional attributes into technical characteristics which will become the requirements for the physical system
8. Establish quantifiable requirements from all above steps
9. Use block diagrams to express interfaces and hardware/software/data relationships for system level
10. Decompose functional requirements/characteristics to lower levels
11. Iterate to test assumptions and reconcile

40