appendix G

SDF Presentation Slides

A System Design and Management Framework

Key Questions in SE Process Development

- What is a system?
- What are the major activities and sub-activities that make up the system engineering process?
- What is the logical sequence of those activities?
- What are the feedbacks within the process and why do they exist?
- How do the various levels within the program hierarchy interrelate?
- How are the major activities decomposed and how do they relate to one another?
- When should iteration occur and how should it be planned for?

Working Definition of "System"

"Any Entity Within Prescribed Boundaries That Performs Work on an Input in Order to Generate an Output"

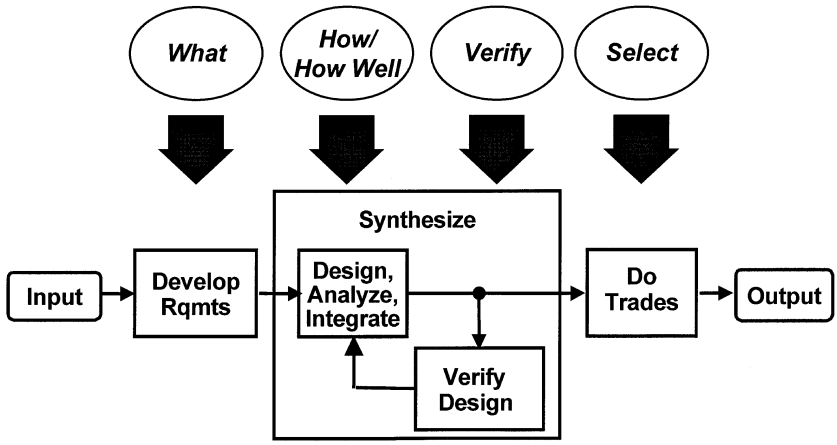
SE Literature Search

Industry Standards
- IEEE-1220
- EIA/IS-632
- Mil-Std-499A
- Army Field Manual 770-786

Individual Authors
- Shinners
- Reinert & Wertz
- Coutinho
- Hall
- Blanchard & Fabrycky
- Chase
- Wymore

Consensus
- What
- How
- How Well
- Verify
- Select

SDF Basic Building Block



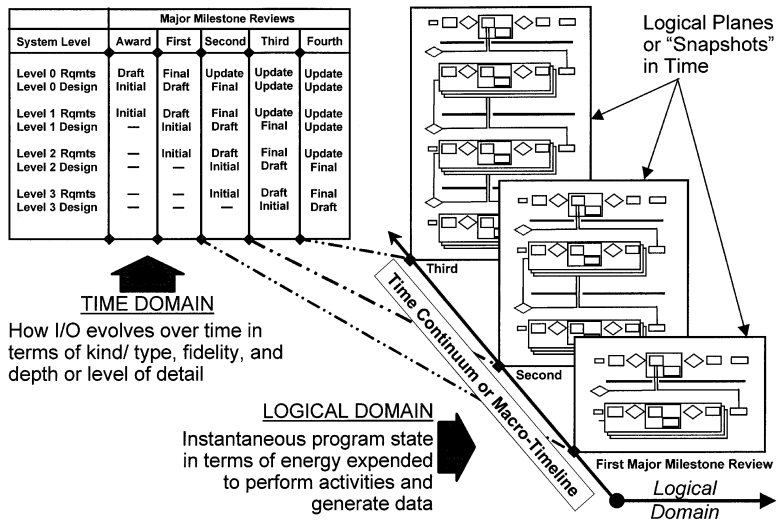
The Time and Logical Domains

Both Time & Logical Views Needed to Clearly Describe Program

- Why have some efforts had limited success in defining a generalized process applicable to many contexts?
  - Time and logical domains not explicitly identified and characterized in distinction
  - When the logical view is overlaid on a chronological view, the resulting process becomes application specific
- When characterized in distinction the overall framework is preserved

Both Views Needed to Provide Full Program Description

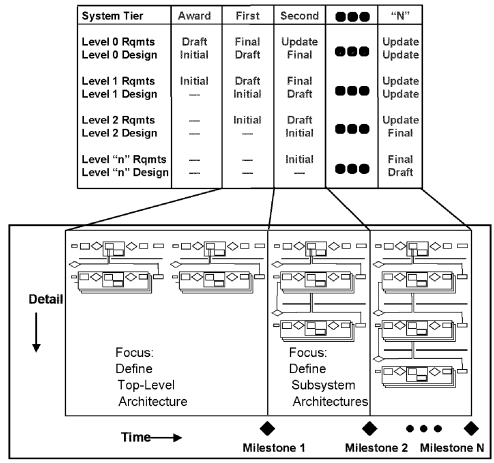
System Level	Major Milestone Reviews				
	Award	First	Second	Third	Fourth
Level 0 Rqmts Level 0 Design	Draft Initial	Final Draft	Update Final	Update Update	Update Update
Level 1 Rqmts Level 1 Design	Initial —	Draft Initial	Final Draft	Update Final	Update Update
Level 2 Rqmts Level 2 Design	— —	Initial —	Draft Initial	Final Draft	Update Final
Level 3 Rqmts Level 3 Design	— —	— —	Initial —	Draft Initial	Final Draft



Time Domain ➔ Output Progression f(time)

- Describes how output evolves over time
- Over time energy is expended in each activity until the desired output at the necessary fidelity is generated
- Two distinct sets of data are generated: Requirements and Design
  - Requirements are developed in sufficient detail such that design activities can be performed with reasonable probability of success ➔ Acceptable Risk
  - The design definition also evolves over time with increasing detail at increasingly lower levels of the system hierarchy.

The SDF in the Time Domain

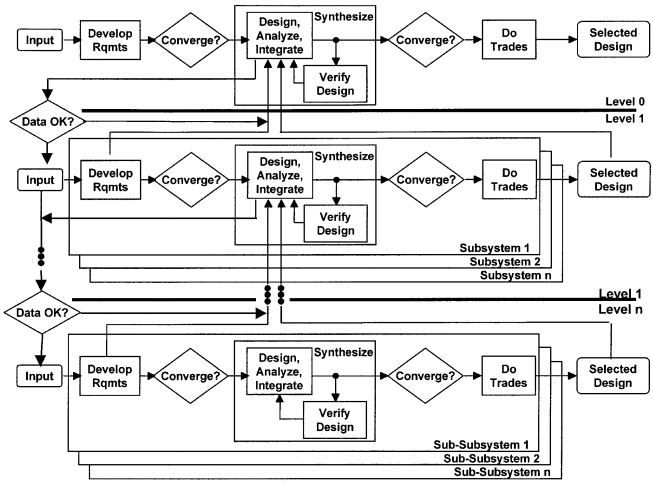


Time Domain Focus: Output as a Function of Time

Logical Domain: Instantaneous Snapshot of Program State

- At any instant in time each activity is performed at some level of intensity at some tier of system hierarchy
- The level of intensity is dependent upon many factors:
  - stability of the input requirements
  - level of complexity of the system
  - whether the system is precedented or not
  - where on the timeline the development effort is occurring
- The time continuum contains an infinite number of "logical planes", each reveals:
  - How many tiers are involved and how many subsystems
  - Logical connections within and between each tier
  - Energy level applied to each activity

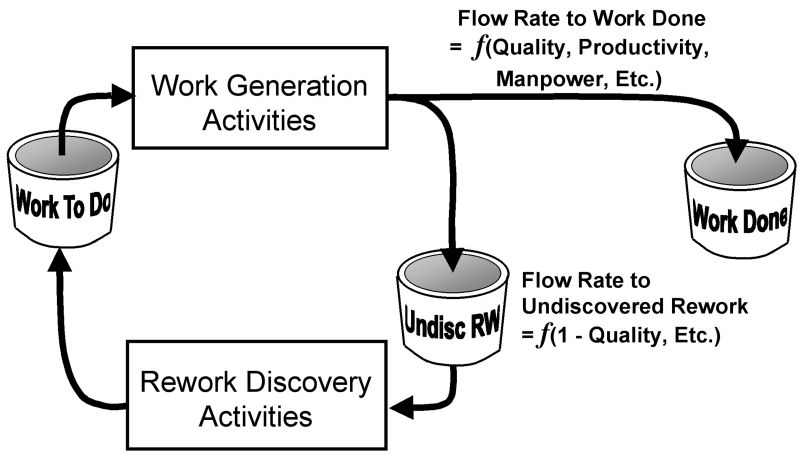
The SDF Logical View



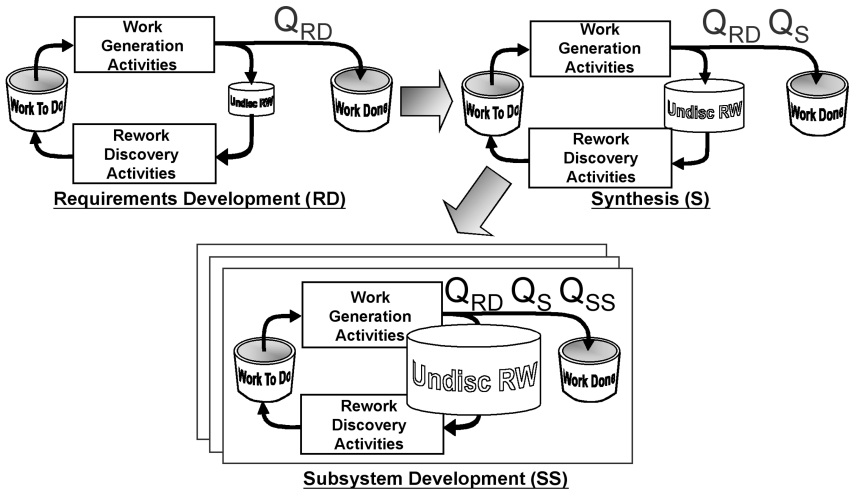
Logical Domain: "Snapshot" of Energy Expenditure

The Rework Cycle

Cost Nemesis—Rework

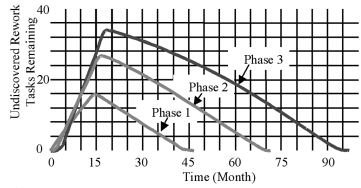


Rework Exponential Growth with Multiple Phases

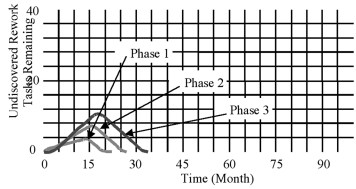


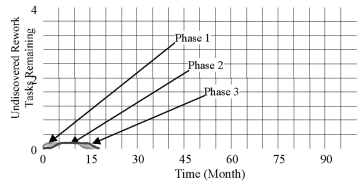
Poor Quality Drives Excessive Rework

Quality = 50%

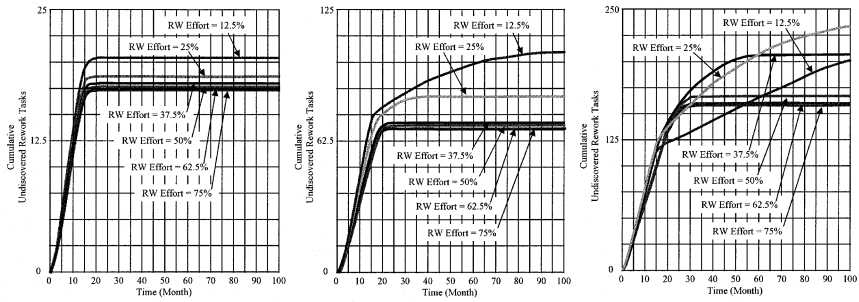


Quality = 70%





Insufficient Effort Focused on Discovering Rework Leads to Still More Rework



Quality=90% Quality=70% Quality=50%

So What's the "So What"?

- Rework (undiscovered & known) is one of the major causes of cost and schedule overruns
- Quality is the greatest leverage point to improve program cost and schedule performance
- Rework Discovery is essential for controlling system development cost and schedule

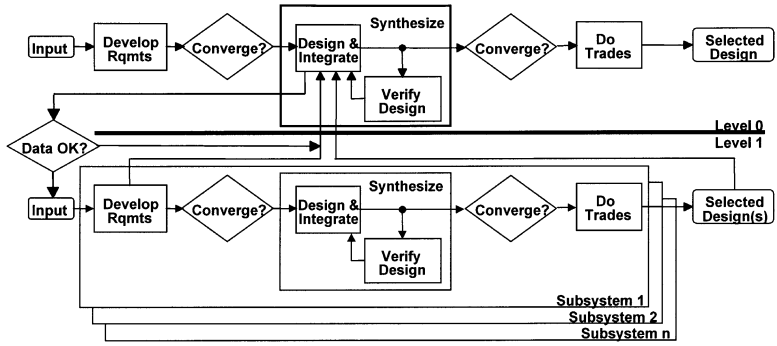
What is the Purpose of Each SDF Activity?

Requirements Development		Design & Analysis				Verification	
Activity	Main Focus	Activity	Main Focus	Activity	Main Focus	Activity	Main Focus
Requirements Analysis	Discover Rework	Identify/Modify Design	Work	Analyze Performance	Discover Rework	Analysis (may be same as Synthesis)	Discover Rework
Mission Analysis	Work	Allocation	Work	Assess Producibility, Testability	Discover Rework	Test	Discover Rework
Rqmts Verification Check	Discover Rework	Functional Decomposition	Work	Optimize	Work	Plan System Test	Discover Rework
Functional Analysis	Work & Discover Rework	Design Integration	Work				

About Half the SDF Activities Focus on Rework Discovery

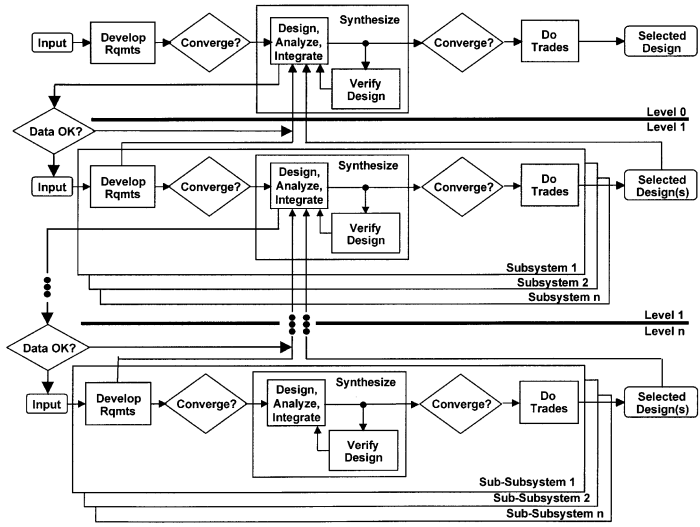
The System Development Framework: Technical

SDF in the Logical Domain



- Same process is used at each product level
- Identifies information flow paths, I/F control responsibility
- Ensures "closed-loop" development tailorable to specific program needs
- Modularity facilitates Tailoring

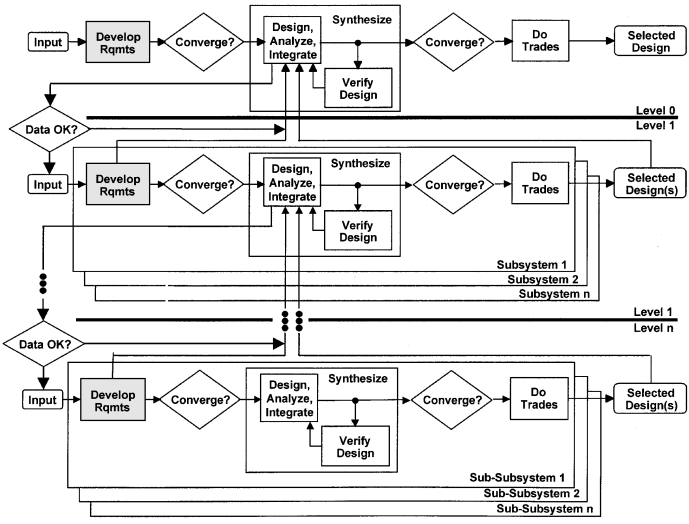
SDF Modular Construction



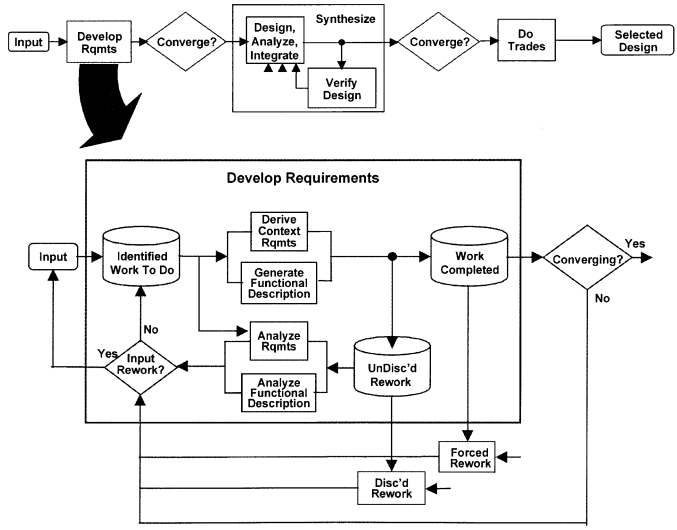
Convergence

- A key criterion in moving from one activity to another is convergence
- Examples Non-Convergence
  - Spacecraft required to communicate with a relay satellite while their orbits preclude such communication
  - When a function cannot be performed without input from another function
  - Unavailability of certain technologies required to satisfy a particular requirements set
  - Stable and consistent requirements but not implementable—at a reasonable cost or schedule
- Occurs when there is an acceptable probability of success that the subsequent activity will converge with that data

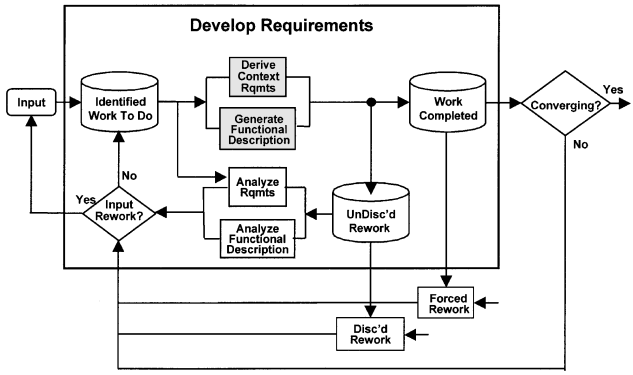
Requirements Development



"Develop Requirements" Activity Decomposed



Requirements Development: Work Activities



INPUT to the Engineering Process

- Customer
  - Immediate
  - Procuring Organization
  - User Community
- Company
  - Corporation
  - Division
  - Department
- Previous Work
  - Business Development
  - Proposal
  - Previous Contract(s)/Studies
- Heritage Products/Designs
  - Division
  - Company
  - Competitor(s)
  - Customer(s)
- New Technologies
  - ITT IR&D, etc.
- Others

Rqmts Originate From Many Sources;
All Must Be Considered To Maximize Success

"What" ➔ *Derive Context Rqmts*

- Determine context in which the system must function over its complete life-cycle
- Identify all mission phases, modes, and states
- Develop mission timeline & Operations Concept
- Identify all external interfaces by mission phase
- Identify critical issues by mission phase (events, technologies, etc.)
- Define environments by mission phase
- **Output**==>Prelim: Derived rqmts, Op's Concept, Context Diagram(s), Entity Relation Diagram(s), Event List(s), external ICDs, FMECA, etc.

First, Identity "What" the System Must Do

Customer-Determined Design

Initial Customer Design

Functions

Identify Major Functions

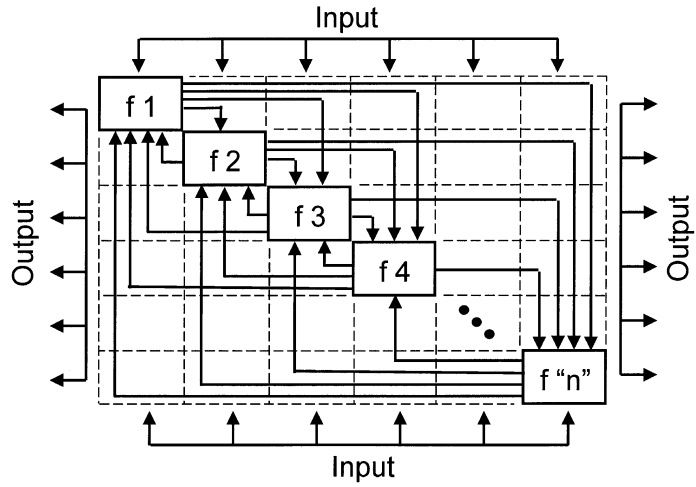
- ⇨Space Segment → ⇨Perform Satellite Operations
  - ⇨Telescope → ⇨Perform Telescope Operations
  - ⇨Spacecraft Bus → ⇨Support Telescope Operations
- ⇨Launch Segment → ⇨Perform Launcher Operations
- ⇨Ground Segment → ⇨Perform Ground Operations

Required Functionality is Derived from the Design Concept

Define Mission Phases

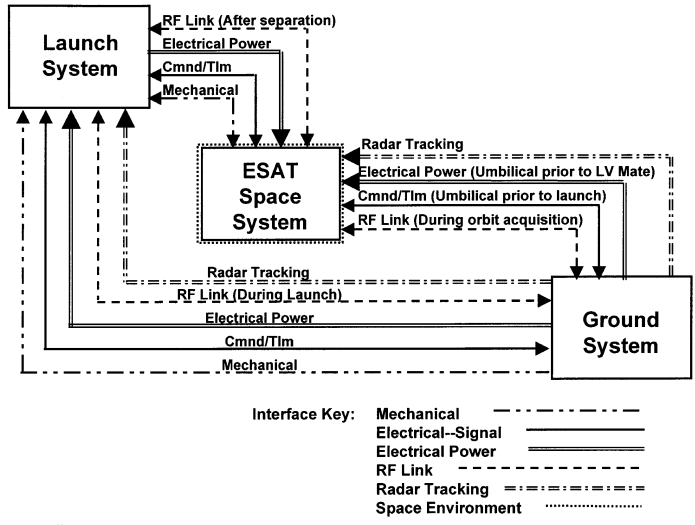
Parameters	Mission Phases				
	Integration & Test	Deploy	Initialization	Operations	Disposal
External Interfaces	Test Fixtures	Launcher Ground Sys	Launcher Ground Sys AKM	Ground Sys Relay Sats Other Sats	Ground System
Environment	Clean Room System Test	Air Ride Van Air Transport Launch site Facilities Fairing	Ascent Trajectory	Operational Orbit	Parking Orbit or Earth Re-Entry
System Modes	Test	Test Launch mode	On-Orbit test Maneuver Appendage Deploy	Nominal Standby Safe Maintenance On-Orbit test	De-Orbit

A Convenient Format*



* Based upon "The N2 Chart", R. Lano, Copyright 1977 TRW Inc.

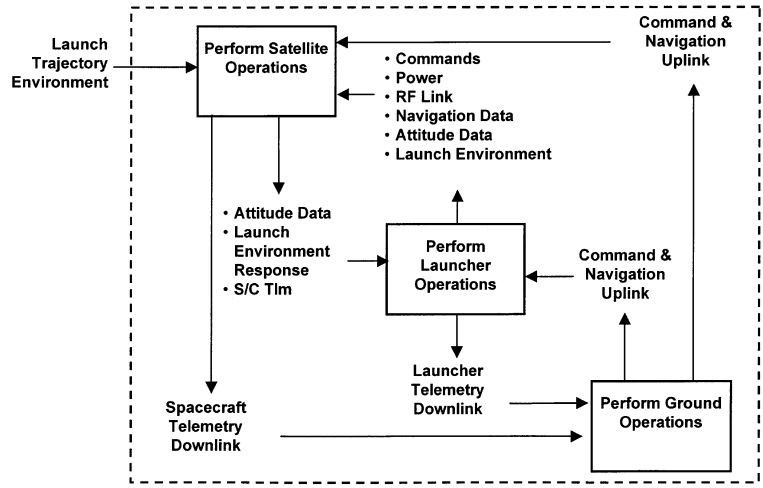
Interfaces--Launch & Orbit Acquisition Phase



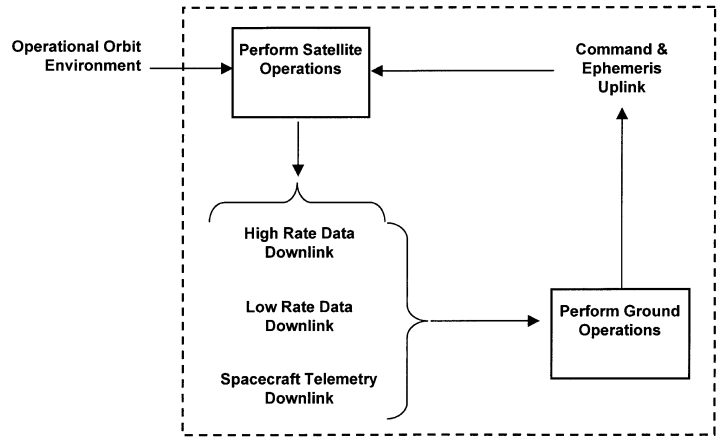
"What" ➔ Generate Funct'l Description

- Identify all functional requirements flowing out of imposed and derived requirements
- Identify performance requirements of each function and the relationships (interfaces, interdependencies, etc.) between functions
- Develop appropriate functional models
- Output ➔ Validated spec(s), functional models (block diagrams, flow diagrams, behavior diagrams, etc.)
- Customer/Stakeholder Consensus

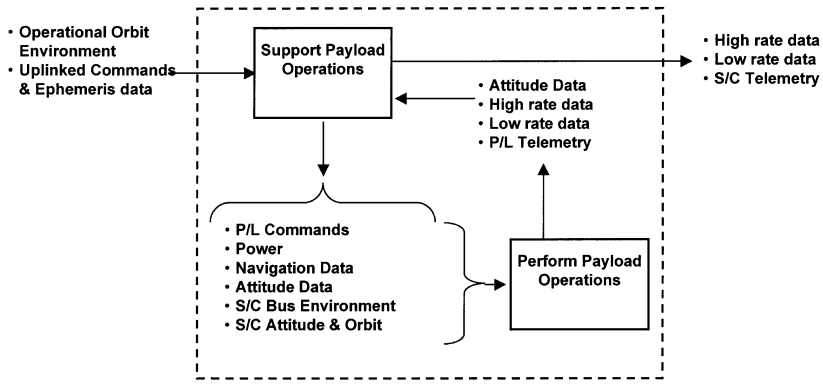
Second, Develop Functional Block Diagrams-- Orbit Acquisition Phase



Operations Phase



Decompose "Perform Satellite Operations" Function by Assuming Separate Bus & Payload



Decompose "Support Payload Operations" by Assuming Standard Bus Subsystems

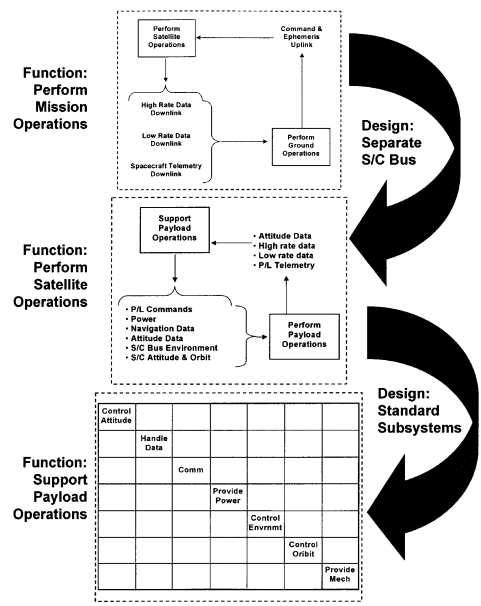
- Operational Orbit Environment
- Uplinked Commands & Ephemeris data
- Attitude Data
- High rate data
- Low rate data
- P/L Telemetry

Control Attitude						
	Handle Data					
		Comm				
			Provide Power			
				Control Envrnmt		
					Control Oribit	
						Provide Mech

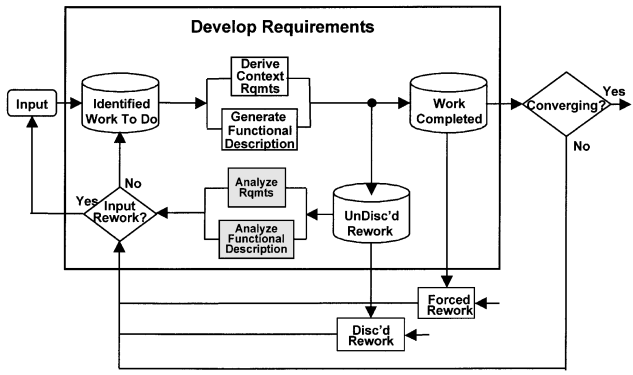
- Attitude Data
- High rate data
- Low rate data
- S/C Telemetry
- P/L Cmnds
- Power
- Navigation Data
- Attitude Data
- S/C Bus Environment
- S/C Attitude & Orbit

Continuity in Decomposition

Each level of decomposition assumes a design concept



Requirements Development: Rework Discovery Activities



"What" ➔ Analyze Requirements

- Identify all requirements, customer desires, customer priorities, constraints
- Analyze for completeness, consistency, etc.
- Interpret Customer rqmts & reach consensus
- Initialize rqmts database & maintain traceability

Output➔"Scrubbed" Requirements Set

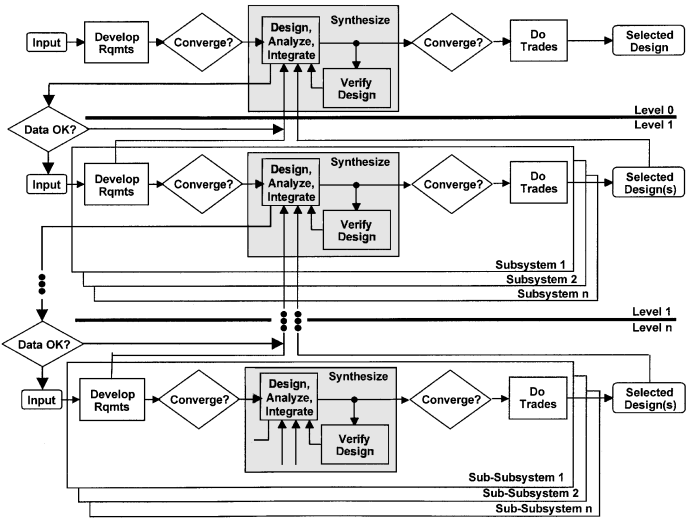
Analyze Rqmts Verifiability

- Determine if all imposed and derived requirements are verifiable
- Determine where in the system build-up each requirement will be verified
- Determine Verification method (Test, analysis, demonstration, simulation, inspection)
- **Output➔**Prelim: Verification Plan/Matrix, verifiable requirements statements

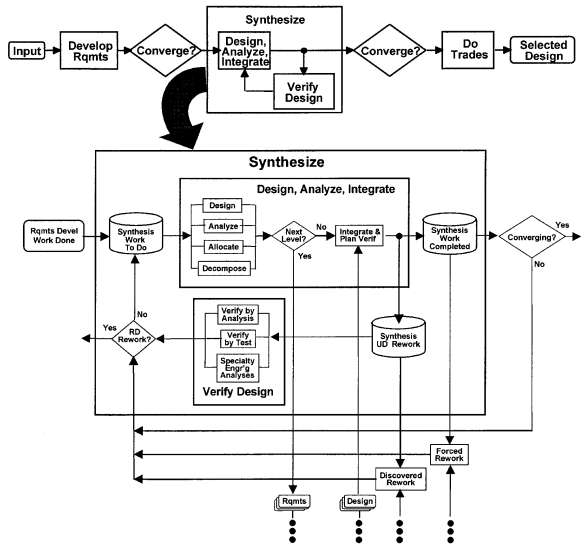
"What" ➔ *Analyze Funct'l Description*

- Verify adequacy of identified functions
  - Performance requirements
  - Output data sufficiency
- Verify interfaces between functions (timing, protocols, data content, etc.)
- Perform appropriate simulations, tests, etc.
- **Output** ➔ Validated spec(s), functional models (simulations, tests, etc.)

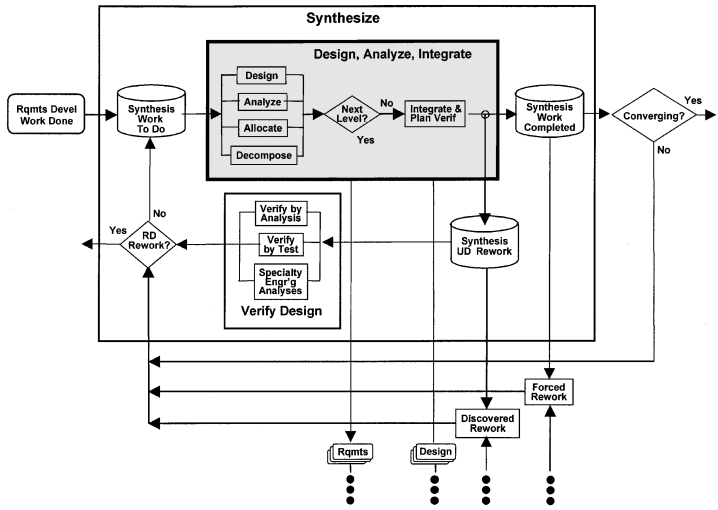
Synthesis



"Synthesize" Activity Decomposed



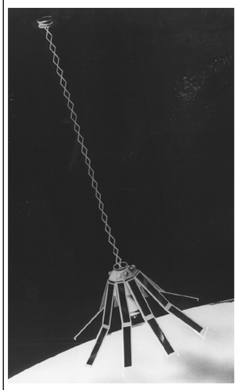
Synthesis Work Activities —Design & Integrate



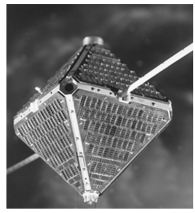
Design

- All Necessary Disciplines Involved
  - –Engineering
  - –Manufacturing
  - –Integration & Test
  - –Operations, etc.
- Quantify Design Space (H/W & S/W)
  - –Parametric Analyses
  - –New Technologies
  - –Heritage Designs
- Generate Preliminary Design
  - – Deployed system and all support equipment
  - – Block diagrams, schematics, drawings, etc.
- Identify and Assess Risk
  - – Technical performance, cost, schedule
  - – Preliminary mitigation approaches
- **Output** ➔ Prelim: H/W & S/W design(s), risk assessment

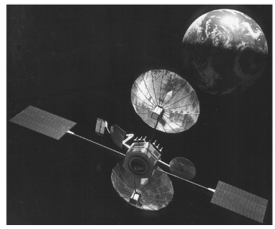
Existing S/C Configurations



GEOSAT
Gravity Gradient
Photo Courtesy JHU/APL

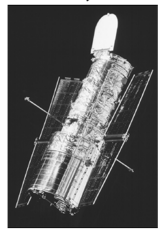


Environmental Research Satellite
No Stabilization
Photo Courtesy TRW, Inc.



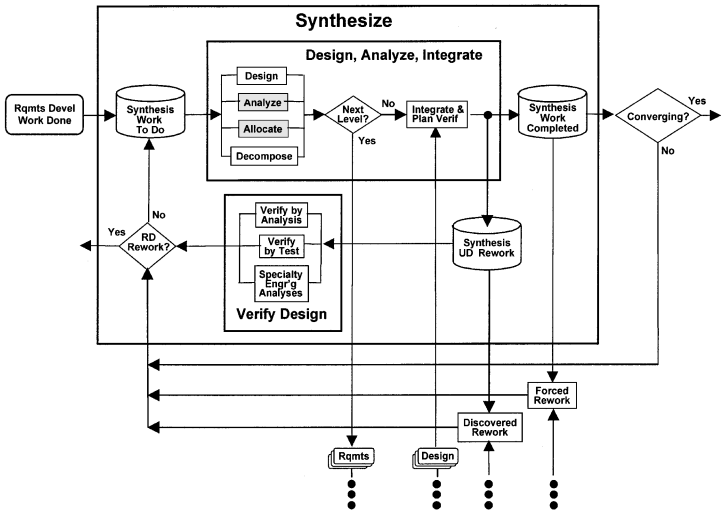
TDRSS
Bias Momentum
Photo Courtesy TRW, Inc.

DSP
Spin Stabilization
Photo Courtesy TRW, Inc.

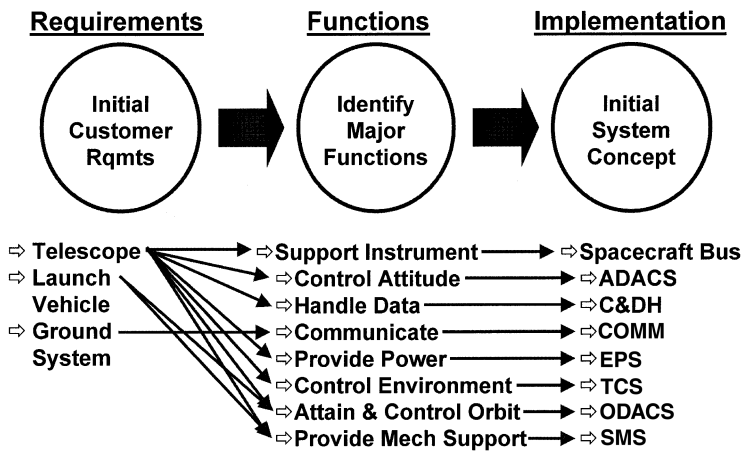


Hubble Space Telescope
Zero Momentum
Photo Courtesy NASA

The "Analyze" and "Allocate" Activities



Allocation of Functionality to Implementation



Analysis

Any and all analyses necessary to quantify design space and parameters

- Mission--Context Definition
- Communications
- Command and Data Handling
- Electrical Power
- Environmental Control
- Propulsion
- Mechanical
- Attitude Determination and Control

Allocation

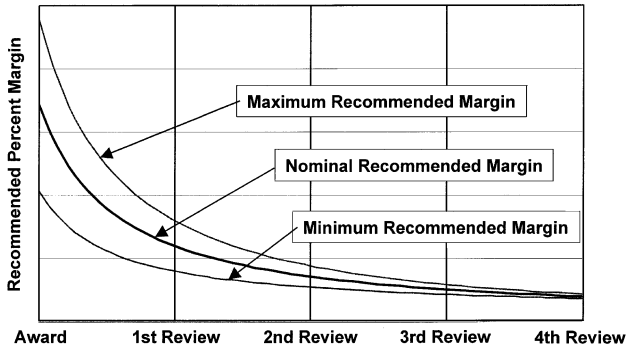
- Allocate functionality, perf, constraints, etc. to system H/W and S/W elements
- Define preliminary budgets
  - Technical: mass, power, throughput, memory, RF links, etc.
  - Cost
  - Schedule
  - Programmatic: Risk, Reliability, Contamination, etc.
- Define Margin and Contingency Rules and Implement in Budgets
- Define/Refine TPMs
- Output ➔ Prelim budgets, TPMs

Allocation ➔ Budgets

	Mass	Power	Memory	Thruput	Etc.
Communications					
Command and Data Handling					
Electrical Power					
Environmental Control					
Propulsion					
Attitude Determination and Control					

Margin Unknown Is Margin Lost!

Notional Convergence of Margin and Reduction in Uncertainty

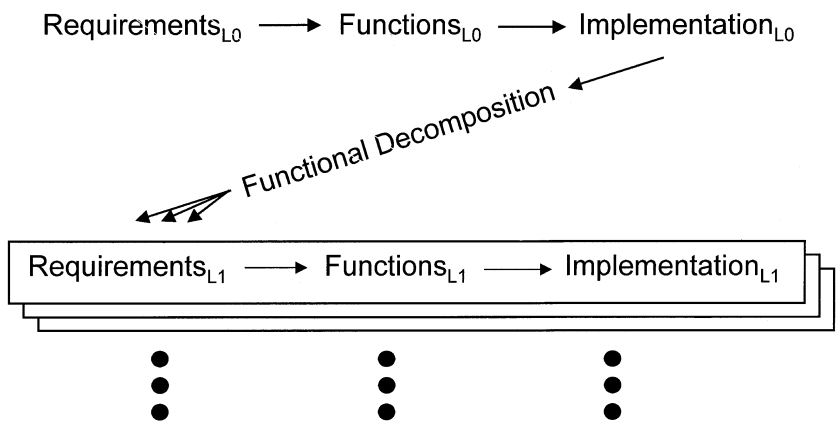


Functional Decomposition Methodology

- Receive function and performance rqmts from RD activity
- Develop design concepts that meet rqmts
- Decompose concept into sub-functions for next-level down activity
- Identify interfaces between sub-functions
- Partition sub-functions into logical groups minimizing interfaces between groups
- Generate functional model and verify
- Generate function and performance rqmts (specs and ICD's) for each group
- Release function and performance rqmts to lower-level activities
- Receive feedback from lower-level development activities and refine specs and ICDs
- Iterate as necessary

Output ➔ Lower level validated specifications, ICD(s), lower level element simulation

Development by Functional Decomposition



Example ➔ Attitude Control Subsystem

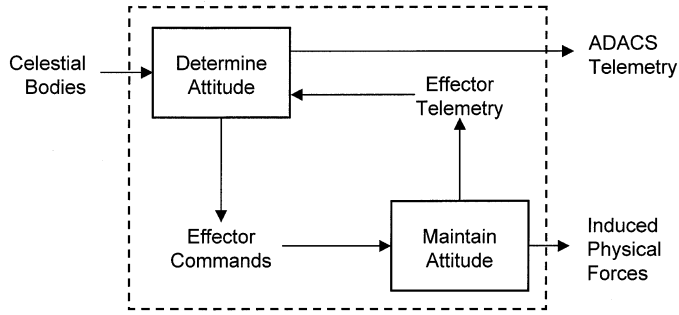
Customer Rqmt	**Required Function**	**Design Candidates**
The spacecraft shall point the Instrument with an accuracy of 0.01 degrees on all three axes	Control Instrument Attitude Required Performance: 0.01 degrees, all three axes	1) Gravity Gradient 2) Spin Stabilization 3) Bias Momentum 4) Zero Momentum

Decomposition:
Derive requirements (sub-functions) needed to implement each design candidate

Gravity Gradient	**Spin Stabilization**	**Bias Momentum**	**Zero Momentum**
• Control Distribution of Mass	• Determine attitude of roll & pitch or yaw axes • Provide Z-axis rotational momentum • Control rotational inertia	• Determine attitude of roll & pitch or yaw axes • Provide Z-axis rotational momentum • Control rotational momentum • De-couple momentum source from S/C Bus	• Determine attitude of all three axes • Control momentum of all three axes

"Control Attitude" Decomposition

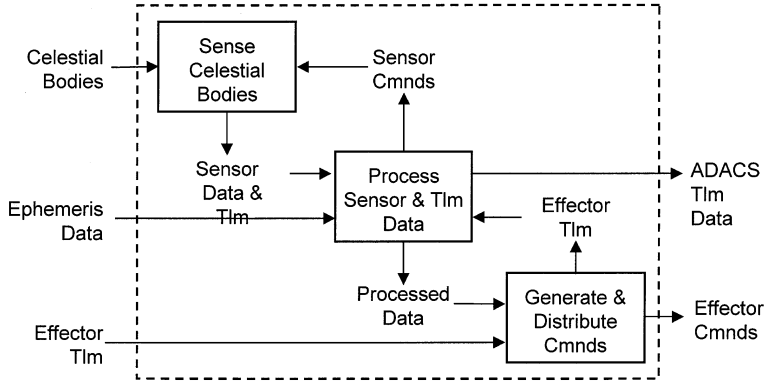
Assumptions: 1) S/C Attitude must be controlled
2) The attitude must be determined onboard the S/C



"Determine Attitude" Decomposed

In order to decompose, at least a concept must be selected

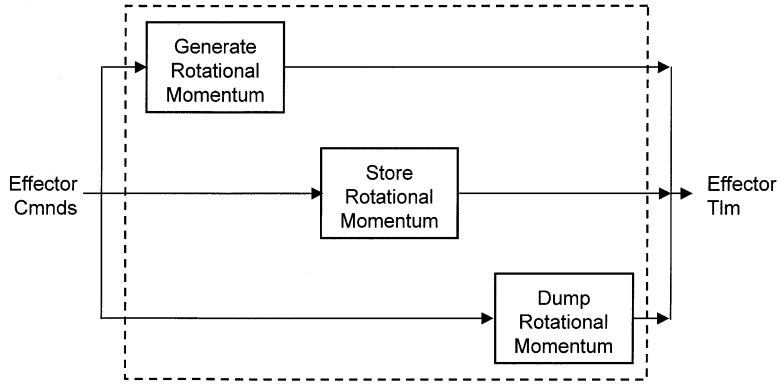
- ⇨Gyros--Dead Reckoning
- ⇨***Celestial Bodies***
- ⇨GPS
- ⇨Ground Uplink
- ⇨Function Not Required



"Maintain Attitude" Decomposed

In order to decompose, at least a concept must be selected

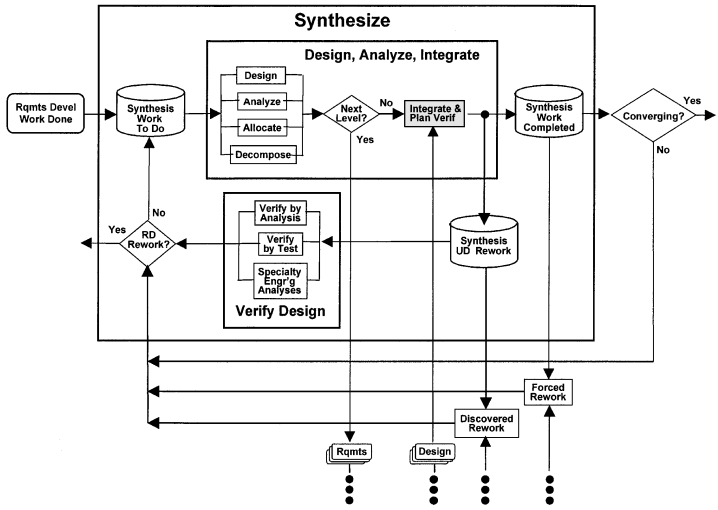
- ⇨Gyros--Dead Reckoning
- ⇨***Celestial Bodies***
- ⇨GPS
- ⇨Ground Uplink
- ⇨Function Not Required



Control Data Flow to Lower Level

- Configuration Control "Design-To" Information for Next-Tier Elements
- **Output** ➔ Configuration controlled documentation

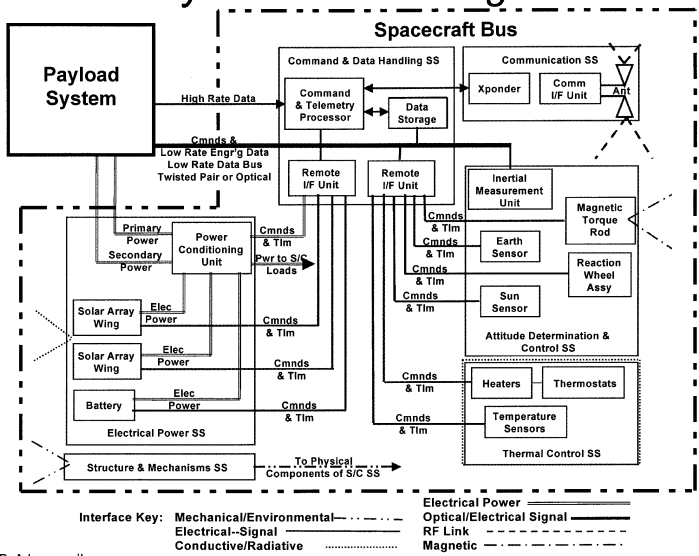
"Integrate & Plan Verification" Activity



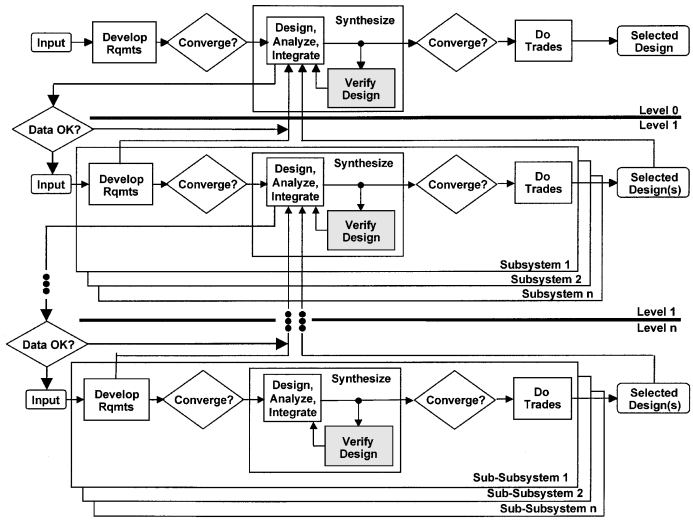
Integrate

- Identify and Characterize I/Fs
- Control Configuration
  - Configuration Control Board
  - Handle Re-allocation
  - Specifications and ICDs
  - Databases, Etc.
- Update Mission Timeline/Operations Concept
- Update Design Definition
  - Block diagrams, schematics, Drawings, layouts
  - budgets, etc.
- Manage Risk
  - Update Risk Identification, Assessment, and Mitigation Approaches
- **Output** → Refined Design

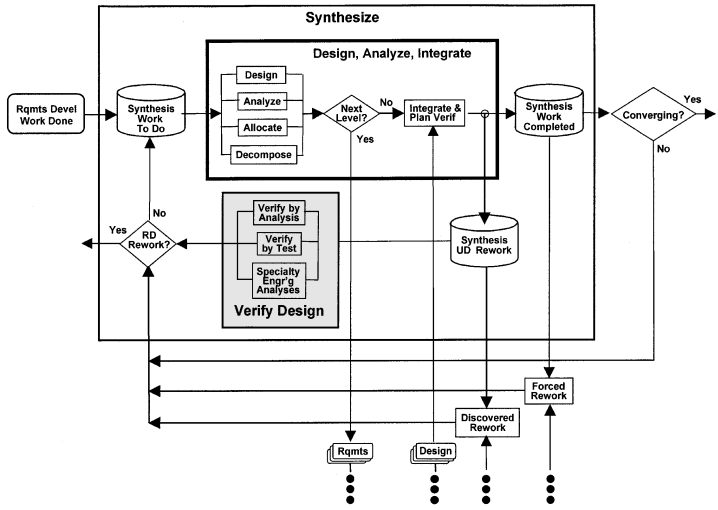
A Notional System Block Diagram



Synthesis—Rework Discovery Activities



Synthesis—Rework Discovery Activities



Verify by Analysis

- Analysis—Analyses to determine “how well” the current design meets requirements
  - In contrast to analyses that define design space as described above in the design activity
- Output ➔ Mission, Electrical, and Mechanical analyses, simulations, etc.

Verify by Test

Design Verification

- Pre-CDR, Integral part of design development activity
- Test Planning
  - –Test requirements
  - –Test flow
  - –Resource planning, Etc.
- Testing
  - –ETMs, prototypes, bread & brassboards, etc.
  - –System Verification Test (SVT)
- Output ➔ Verified Design

Product Verification

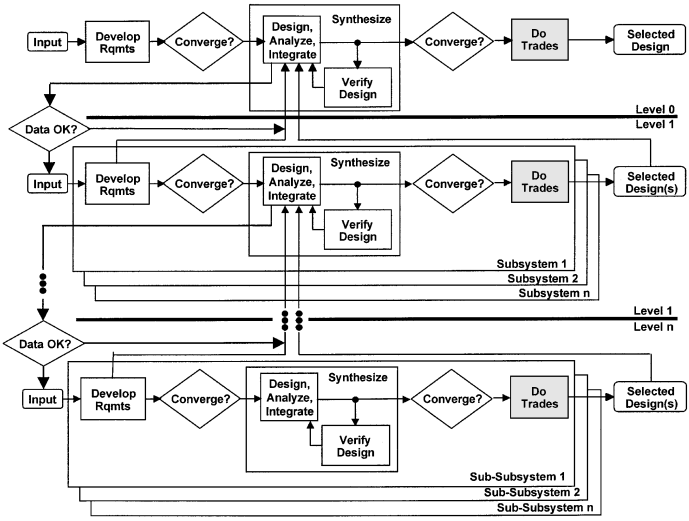
- Post CDR
- Focuses on the product to be deployed
- Test, analysis, simulation, demonstration, inspection
- Output ➔ Verified Product

Verification Activities Begin Early In The Program

Specialty Engineering

- Is the design testable within resource and time constraints?
- Is the design producible within resource and time constraints?
- Is the design acceptable with respect to
  - –EMI/EMC
  - –Reliability
  - –Maintainability, affordability, supportability
  - –etc.

Trade Analysis

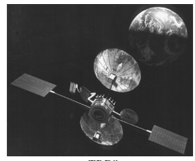


Select

- Define Trade Criteria
  - Technical, Cost, Schedule, Risk
  - System Robustness
  - Sensitivity Analyses
- Assemble Trade Matrix
  - Must Have's, Wants, Utility
  - Value/Impact
- Assess Each Candidate
  - Select Best Design Values
- Selection
  - If multiple candidates are compliant and equally acceptable to design team, make selection at tier above
  - If selection is clear at the current tier, make selection
- Output ➔ Selected Design
- Rigor defined by program need

Trades Occur After Criteria Is Developed;
The Literature Provides Many Methodologies

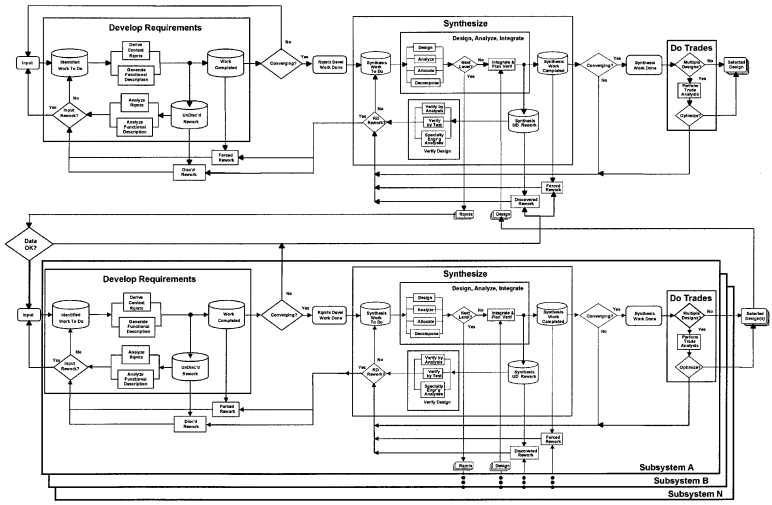
ADACS Candidate Technical Assessment

Architecture				
Gravity Gradient	**Spin Stabilization**		**Bias Momentum**	**Zero Momentum**
				
GEOSAT *Photo Courtesy APL JHU*	**DSP Spacecraft** *Photo Courtesy TRW*		**TDRS** *Photo Courtesy TRW*	**Hubble Space Telescope** *Photo Courtesy NASA*
Passive Mass Dist	**Aspect ratio Mass Balance Earth Sensor IMU, OBC Thrusters Mag Torqures**	**Design Elements**	**Earth Sensor Sun Sensor IMU, OBC Thrusters Momentum Wheel Mag Torquers**	**Earth Sensor Sun Sensor Star Tracker IMU, OBC Thrusters Reaction Wheels Mag Torquers**
± 5° two axes	**± 0.1° to ± 1°**	**Accuracy**	**± 0.1° to ± 1°**	**± 0.001° to ± 1°**
Cannot meet 0.01 accuracy rqmt	**Spin not OK Cannot meet 0.01 accuracy rqmt**	**Assessment**	**Cannot meet 0.01 accuracy rqmt**	**Meets all rqmts**

What About Optimization?

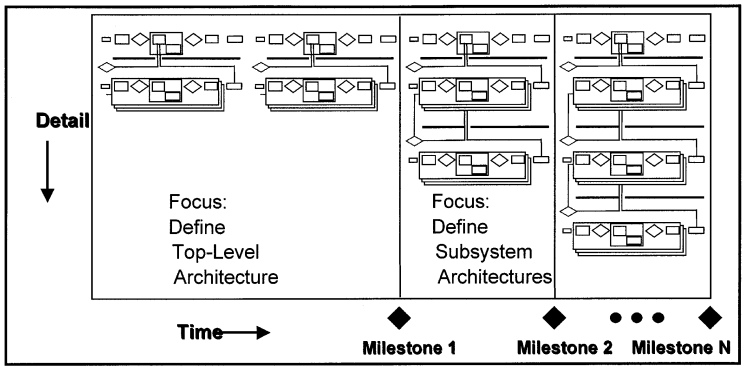
- It is not our purpose to discuss the myriad and specialized techniques
- It is our purpose is to describe where optimization occurs and how it impacts the overall process
- Optimization, by definition, implies a change to the design, therefore, the SDF provides feedback for it
- To some degree optimization occurs within each activity
- It is explicitly addressed here because it is at this point that technical, cost, and schedule criteria are available
- Elsewhere in the process, each design is being developed so optimization occurs through the iterations that naturally occur

SDF 2nd Level Decomposition



See page 90 for larger figure.

The SDF In The Time Domain



- One iteration of the EP may take “seconds” or much longer
- Incremental Solidification provides a mechanism for managing risk
- Outputs are defined as *f*(timeline)
- The SDF is iterated as required

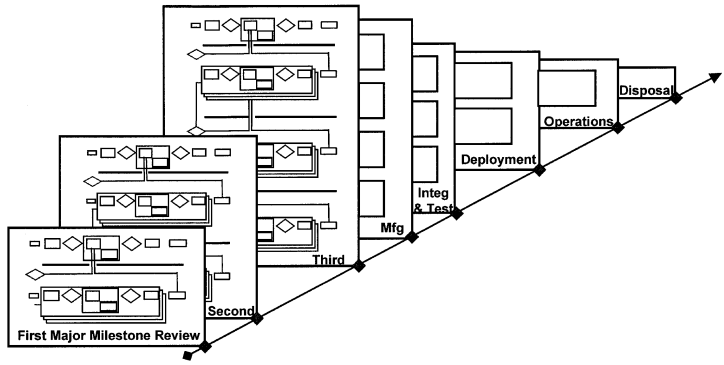
Time-Phased Output at Planned Fidelity

System Tier	Award	First	Second	● ● ●	"N"
Level 0 Rqmts	Draft	Final	Update	● ● ●	Update
Level 0 Design	Initial	Draft	Final		Update
Level 1 Rqmts	Initial	Draft	Final	● ● ●	Update
Level 1 Design	—	Initial	Draft		Update
Level 2 Rqmts	—	Initial	Draft	● ● ●	Update
Level 2 Design	—	—	Initial		Final
Level "n" Rqmts	—	—	Initial	● ● ●	Final
Level "n" Design	—	—	—		Draft

Full Life-Cycle

- Each mission phase imposes unique requirements on the system
- In order to maximize success, these requirements must be considered from the start
- The design effort generally continues up to the Critical Design Review
- After CDR, program moves from significant design effort to mfg, integration and test, deployment, op's, and disposal
- In production programs, provide feedback to the design activity capturing lessons learned from the deployed systems

Full System Life-Cycle



The Full System Life-Cycle must be Considered at the Earliest Stages of Development

Tailorability Of The SDF

- We have established that System Development Framework is the same for each level
- While not all SDF activities represent significant effort in every situation; generally, all are performed to some level of fidelity
- Therefore, tailoring is ***not*** done by changing the SDF, but by:
  - Effective partitioning of system elements (DSM)
  - Modulating the kinds and extent of documentation required
  - Modulating the level of detail and the scope of the activities performed

The System Development Framework:

Managerial

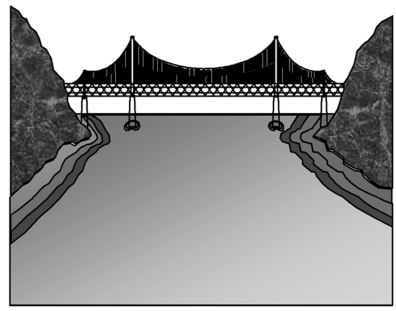
Key Questions

- How should information flow and who is responsible for which interfaces?
- How should the Managerial aspects of system development be structured?
- How should the technical and managerial activities be coupled?

What Are Mgmt & Technical Activities?

Managerial/Programmatic

- Configuration Management
- Risk Management
- Cost Management
- Schedule Management
- Customer Interface
- Subcontracts Management
- Etc.

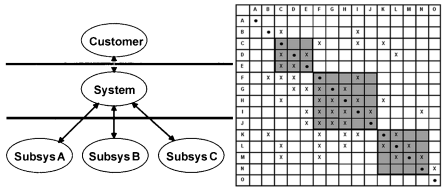


Technical

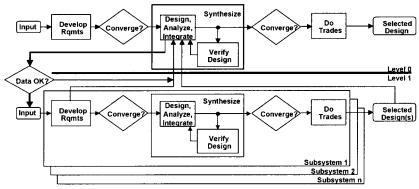
- Requirements Development
- Synthesis
  - Design
  - Analysis
  - Integration
  - Verification
- Trades

These activities are closely coupled;
How they are coupled can be defined by the SDF!

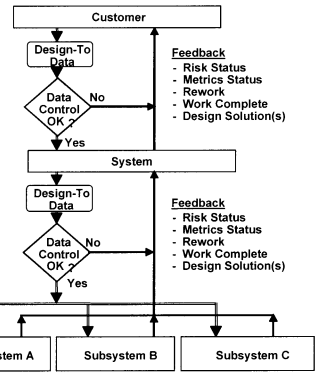
Program Structure and Control



Start with Current Design, Partition elements such that the number of interfaces is minimized



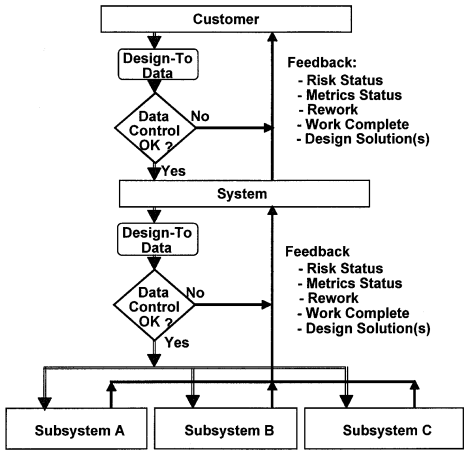
System Development Framework



Resulting Team Structure & Control

Organize according to Architecture
Operate according to SDF Control Logic

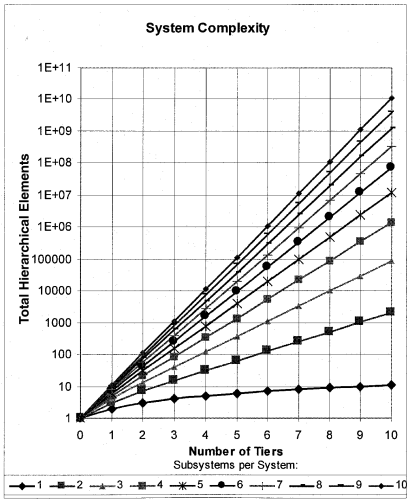
The SDF Defines IPT Interrelationships



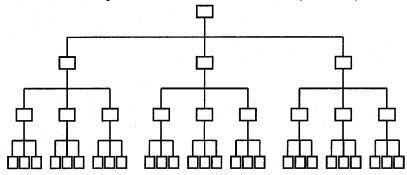
- IPTs function independently within established bounds
- Configuration Management
  - Change Impact
  - Database Structure
  - TPMs, Budgets
- Risk Management
  - Periodic Review
- Roles & Responsiblities
- IPT Interfaces
  - ID, Characterization, Cntl
- Cost Management

"Programmatic" I/Fs are defined by the SDF

Exponential Growth in Complexity



An Example: 3 subsystems per system, Find number of Total System Elements (TSE)



S = Number of Subsystems per System

m = Total number of Hierarchical levels

$$TSE = \sum_{n=0}^{m} S^n = 40$$

Some SDF-Derived Principles

Some Principles

- Acceptable risk is a key criterion in deciding to move from one activity to the next—The difficulty is accurately quantifying it
- Risk cannot be managed if it has not been identified and/or understood
- In conceptual architecting, the level of detail needed is defined by the confidence level desired
- A function cannot be decomposed. Only implementation can be decomposed. That which allows decomposition is knowledge about implementation

More Principles

- Process understanding is no substitute for technical understanding—It is the technical understanding that enables development by decomposition
- Before a process can be improved it must be described
- Given our definition of "system", the same System Development Framework can be used at any tier of design development
- Costs due to rework increase exponentially with time

Suggestions for Implementation in Industry—Focus on Output

- The SDF should be “Tailored” by identifying up front required inputs and outputs for each SDF activity
- Develop Exit Criteria for each review—These are derived directly from the outputs identified in the SDF
- Define required fidelity or “completeness” as a function of the program timeline.
- For each review, the program must produce the generalized SDF output—The structured approach is followed by default