CMPE349 Intro to Professional Practice

Architectural View

Functional Trade Studies

- **Trade studies are hard to do at the functional level**
- **Heuristic: Do the hard parts first**
- **What do we trade?**
  - **Function sets defining different mission scenarios**
  - **Different functional decompositions of the same higher-level function**
- **Alternatives should span the range of potential solutions**
- **What are our metrics?**
  - **Number**
  - **Complexity**
  - **Reuse**
  - **Scaleability**
  - **Integrity/reliability**
- **The trade need not be explicit or detailed, but does need to be captured.**
- **Heuristic: Simplify, Simplify, Simplify!**
- **Heuristic: It doesn’t exist if it isn’t written down**

Functional/architectural tradeoff example

0, 1, 3, 9

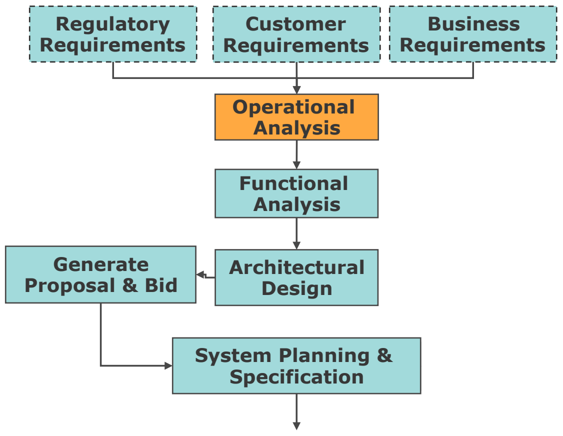
1,2, 3,4,5

		Functional Product Requirements - Operating Parameters						
Maximize, minimize, or target								
Customer Need	Importance Rating	Arch #1	Arch #2	Arch #3	Arch #4	Arch #5	Arch #6	Arch #7
ADS-B functionality	5	9	9	9	9	9	9	9
Advanced Surveillance Functionality	3	9	9	9	9	9	9	9
DSB-AM Voice	5	9	9	9	9	9	9	9
ATS Data Comm.	3	9	9	9	9	9	9	9
Good availability	3	9	3	0	1	3	9	3
Good continuity	4	9	3	0	1	3	9	3
Minimize interference to existing Sys.	3	3	3	3	3	1	3	1
Minimize susceptibility to existing Sys.	3	3	3	3	3	1	3	1
Upgrade of existing radio	1	0	0	0	0	0	0	0
Minimize Number of Antennas	4	0	0	3	3	9	1	9
Comp. wcurrent antenna placement	2	1	1	3	3	9	3	9
AOC Data Comm.	3	9	9	9	9	9	9	9
Minimize acq. cost.	5	0	3	1	0	3	1	1
Minimize installation complexity	3	1	3	3	3	9	1	1
Relative Importance		257	236	221	223	294	270	260
Target Range								

Things to consider

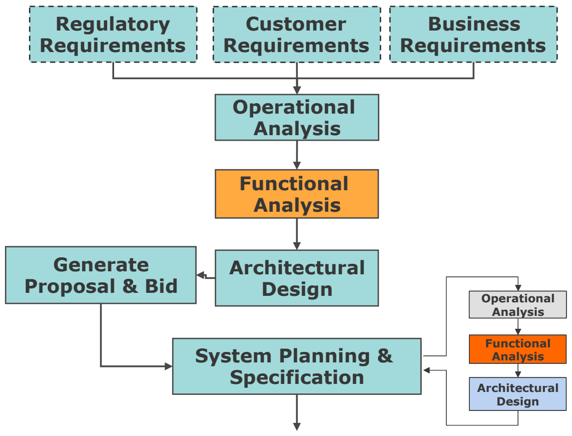
- **Heuristic**: Act on fact!
- Often the simple trade study will only show what you don’t know.
- **Heuristic**: Know what you don’t know.
- Be careful with your weighting, as it can change the results of table-based trades…
- …try slight modifications of the weighting to check for sensitivity.
- Less sensitivity to weighting is good!
- **Heuristic**: The last time your solution is perfect is before you show it to someone else.

Operational Analysis review



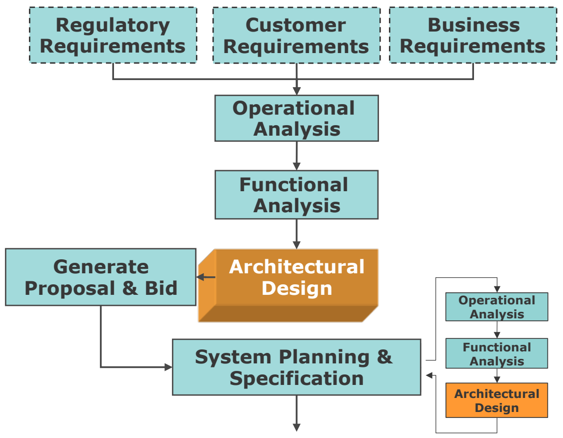
- What is the **scope** of work of the system?
- What are the **lifecycle effects**
- What are the **missions**
- What do we know about the **interfaces**
- What doesn't the **customer/consumer/ client/caretaker** know or understand?

Functional Analysis Overview



- **Define** what the system must do to accomplish the work.
- **Translate** operational requirements into a functional model.
- **Output** functional requirements that constrain the behavior of the model.
- These will be the **system** requirements

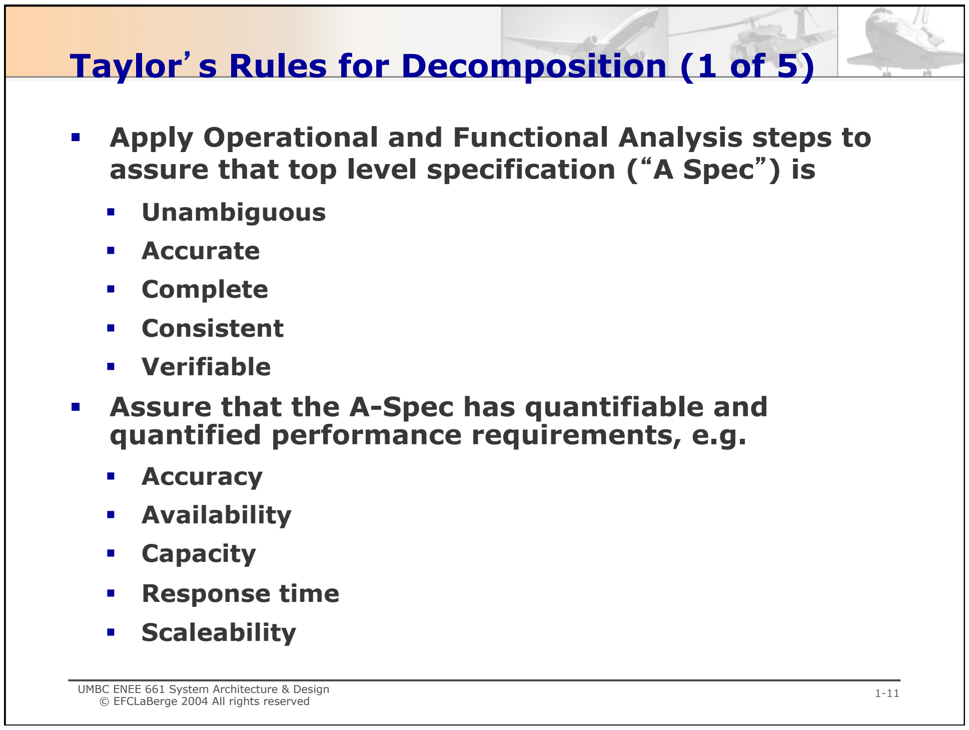
Architectural Design Overview



- Create the **physical model** of the system
- **How** the system implements the requirements
- **Defines** hardware and software components
- **Defines** the interfaces that connect components
- **Defines** the behavior of the components.
- **Allocates** requirements to Configuration Items (CIs)

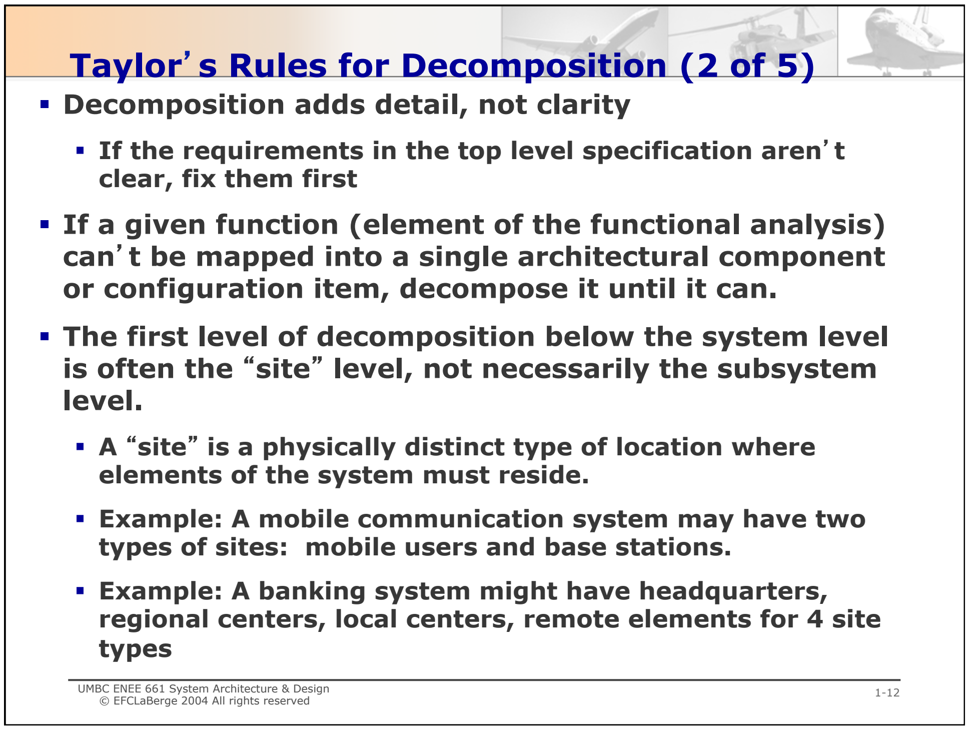
Design is less conceptual than *analysis*

- **Heuristic** (paraphrase): There comes a time to shoot all the analysts and get around to building something.
- **Heuristic**: Avoid the "analysis to paralysis" syndrome.
- The design **problem** is "**where to start?**"
- The design **weakness** is "I already know how to do this!"



Taylor's Rules for Decomposition (1 of 5)

- Apply Operational and Functional Analysis steps to assure that top level specification ("A Spec") is
  - Unambiguous
  - Accurate
  - Complete
  - Consistent
  - Verifiable
- Assure that the A-Spec has quantifiable and quantified performance requirements, e.g.
  - Accuracy
  - Availability
  - Capacity
  - Response time
  - Scaleability



Taylor's Rules for Decomposition (2 of 5)

- Decomposition adds detail, not clarity
  - If the requirements in the top level specification aren't clear, fix them first
- If a given function (element of the functional analysis) can't be mapped into a single architectural component or configuration item, decompose it until it can.
- The first level of decomposition below the system level is often the "site" level, not necessarily the subsystem level.
  - A "site" is a physically distinct type of location where elements of the system must reside.
  - Example: A mobile communication system may have two types of sites: mobile users and base stations.
  - Example: A banking system might have headquarters, regional centers, local centers, remote elements for 4 site types

Taylor's Rules of Decomposition (3 of 5)

- Allocate functions to a single site type
  - If the function can't be allocated to a single site type, then decompose it!
  - A function that can be allocated to a single site type need not be decomposed at this time.
  - This decomposition is the "B Spec"
  - Update the functional analysis to match any additional decompositions done in this step
- A Configuration Item is the instantiation of a group of similar functions that are
  - Relatively autonomous from other functions
  - Can be developed by a single team of 3-8 developers
- Reapply the functional allocation and decomposition rules to the CIs

Taylor's Rules of Decomposition (4 of 5)

- Most systems continue the decomposition to the component level
  - This is the level where COTS or pre-existing products come in
  - The component spec (functional, performance, etc) is the "C Spec".
- Update the functional analysis! DO IT!
- Each leaf in the functional architecture must map to a single CI.
  - Multiple functions to one CI is fine.
  - Single leaf function to multiple CIs is a recipe for disaster.
  - Stop the decomposition when this rule is met!

Taylor's Rules of Decomposition (5 of 5)

- **How to handle performance requirements?**
  - **If performance can be allocated to single CI, handle like functional requirement.**
  - **If not, performance must be partitioned!**
- **Maintain a quantitative assessment of the performance partitioning**
  - **Technical Budgeting (we'll discuss later in detail)**
- **Heuristic: It doesn't exist if it isn't written down!**

Two Types of Physical Architectures

Generic Architecture

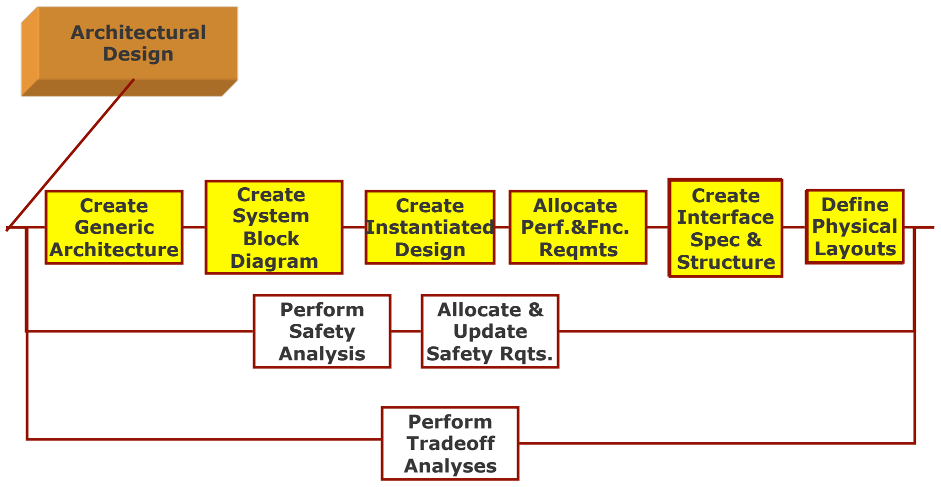
- **Contains components that support all required functions**
- **Includes support resources for operation, maintenance, distribution, training, etc.**
- **Does not specify performance characteristics of each physical resource.**
- **Contains components that can be implemented multiple ways.**

Instantiated Architecture

- **Starts with generic architecture**
- **Includes complete definitions of performance characteristics**
- **Allocates specific COTS products or custom development to components**
- **Identifies quantities of each component**

Heuristic: Except for good and sufficient reasons, generic and instantiated structuring should match

The architectural design process


Architectural Design
Create Generic Architecture
Create System Block Diagram
Create Instantiated Design
Allocate Perf.&Fnc. Reqmts
Create Interface Spec & Structure
Define Physical Layouts
Perform Safety Analysis
Allocate & Update Safety Rqts.
Perform Tradeoff Analyses

Architectural Heuristic

- "When working on a problem, I never think about beauty. I think only of how to solve the problem. But when I have finished, if the solution is not beautiful, I know that it is wrong."

Buckminster Fuller

Cautions related to the physical architecture

- **There is cultural inertia to jump immediately to this step based on perceived prior experience**
  - **Local optimizations in the architecture**
  - **Reduce the effectiveness and/or increase the cost of the final implementation**
- **Successful architectural design requires iteration.**
- **Consider multiple options.**
- **Be careful not to let pre-conceived ideas drive architectural decisions.**
- **Interface definition is critical**
  - **Spend the time required to complete this step.**
  - **It will reduce the need for future rework.**

Architectural Heuristic

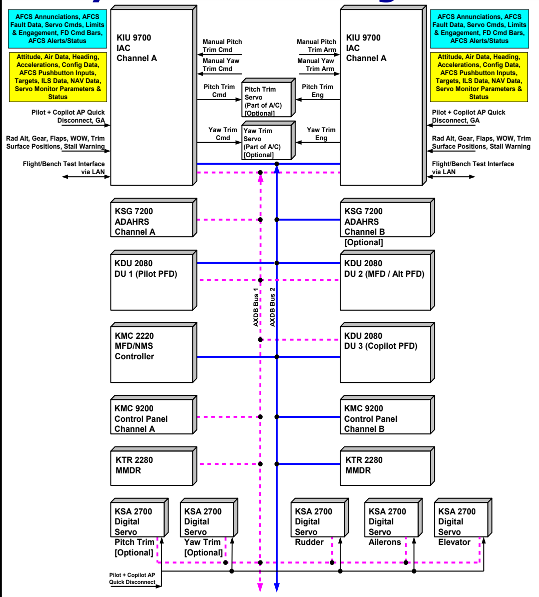
- **Employ the principle of "deferred commitment"**
- **The later that you can commit to a specific implementation technology, the more robust the final architecture will be.**
- **Make a implementation technology choice when you must, not when you can.**

Configuration Item Matrix

- Brainstorming activity results in list of CIs
- In some cases, the customer will have specified configuration items as part of the requirements. Capture this as part of the Architectural Design step.

		Functions				
		Navigation	Lateral Guidance	Vertical guidance	Threat Avoidance	Mission Planning
Configuration Items	ADC #1	X		X		
	ADC #2	X		X		
	CDU #1	X				
	CDU #2	X				
	CMFD#1		X	X		
	CMFD#2		X	X		
	EFIS #1			X		
	EFIS #2			X		
	EGI #1	X		X		
	EGI #2	X		X		
	IFF	X		X		
	ILS	X	X	X		
	MISP #1	X	X	X	X	X
	MISP #2	X	X	X	X	X
	Radar Alt	X		X		
	CSDS				X	X
	MMU					X

System Block Diagram


AFCS Annunciations, AFCS Fault Data, Servo Cmds, Limits & Engagement, FD Cmd Bars, AFCS Alerts/Status
Attitude, Air Data, Heading, Accelerations, Config Data, AFCS Pushbutton Inputs, Targets, ILS Data, NAV Data, Servo Monitor Parameters & Status
Pilot + Copilot AP Quick Disconnect, GA
Rad Alt, Gear, Flaps, WOW, Trim Surface Positions, Stall Warning
Flight/Bench Test Interface via LAN
KIU 9700 IAC Channel A
Manual Pitch Trim Cmd
Manual Yaw Trim Cmd
Pitch Trim Cmd
Pitch Trim Servo (Part of A/C) [Optional]
Yaw Trim Cmd
Yaw Trim Servo (Part of A/C) [Optional]
Manual Pitch Trim Arm
Manual Yaw Trim Arm
Pitch Trim Eng
Yaw Trim Eng
KIU 9700 IAC Channel A
AFCS Annunciations, AFCS Fault Data, Servo Cmds, Limits & Engagement, FD Cmd Bars, AFCS Alerts/Status
Attitude, Air Data, Heading, Accelerations, Config Data, AFCS Pushbutton Inputs, Targets, ILS Data, NAV Data, Servo Monitor Parameters & Status
Pilot + Copilot AP Quick Disconnect, GA
Rad Alt, Gear, Flaps, WOW, Trim Surface Positions, Stall Warning
Flight/Bench Test Interface via LAN
KSG 7200 ADAHRS Channel A
KSG 7200 ADAHRS Channel B [Optional]
KDU 2080 DU 1 (Pilot PFD)
KDU 2080 DU 2 (MFD / Alt PFD)
AXDB Bus 1
AXDB Bus 2
KMC 2220 MFD/NMS Controller
KDU 2080 DU 3 (Copilot PFD)
KMC 9200 Control Panel Channel A
KMC 9200 Control Panel Channel B
KTR 2280 MMDR
KTR 2280 MMDR
KSA 2700 Digital Servo Pitch Trim [Optional]
KSA 2700 Digital Servo Yaw Trim [Optional]
KSA 2700 Digital Servo Rudder
KSA 2700 Digital Servo Ailerons
KSA 2700 Digital Servo Elevator
Pilot + Copilot AP Quick Disconnect

- You may violate the 7+/-2 rule!
- Use previous results
  - System Boundary Diagram
  - Data Flow Diagram
  - Control Flow Diagram
- Show the connectivity between CIs first
- Assign preliminary interconnection types, i.e., analog, digital, discrete
- Evaluate interconnects based on performance
- Show unidirectional vs. bidirectional interfaces.
- Establish point-to-point & bus interconnections