

Part 1: What does my customer want?

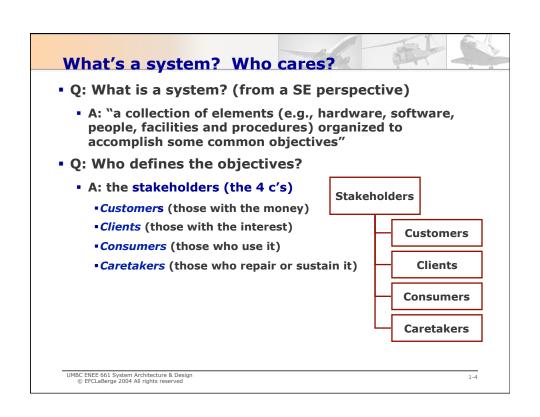
UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved 4 4

Let's start at the very beginning

- How do I have any idea what it is I'm supposed to build?
- Do not skip this step...
- ...otherwise you're running around building/ designing something (which is fun)...
- ...with no idea at all when you will be done...
- ...or if your work will be useful.
- You are an not an artist (not a bad thing).
- ...you're an engineer, so your product must be useful!

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved

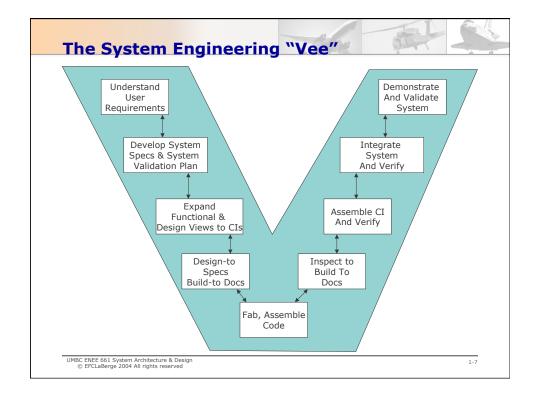
LaBerge's Law #3 • Every engineering problem is a design problem. • In fact designing solutions is what engineering do! What are the requirements? What do I know? What do I need to find out or compute? No What's my solution? Does my solution meet the requirements? Document it!



You don't get paid until you're complete (and successful)

- What determines success?
- Heuristic: Success is defined by the stakeholder, not the architect.
- Heuristic: A system engineer (architect) must maintain a broad perspective of the problem.
- Heuristic (Aristotle): In order to understand anything, you must not try to understand everything

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved



Systems Engineering is a fractal process

- Fractal
 - Fractals are non-regular geometric shapes that have the same degree of non-regularity on all scales.
- Heuristic: One person's architecture is another person's detail. One person's system is another person's component.
- Heuristic: In general, each system level provides a context for the level(s) below
- Heuristic: Leave the specialties to the specialist. The level of detail required by the architect is only to the depth of an element or component critical to the system as a whole.
- Heuristic: The successful system architect or engineering understands the rudiments of all of the external systems and interfaces.

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved 1-8

Create graphical artifacts first!

Create graphics first, text second when depicting artifacts of a system design.



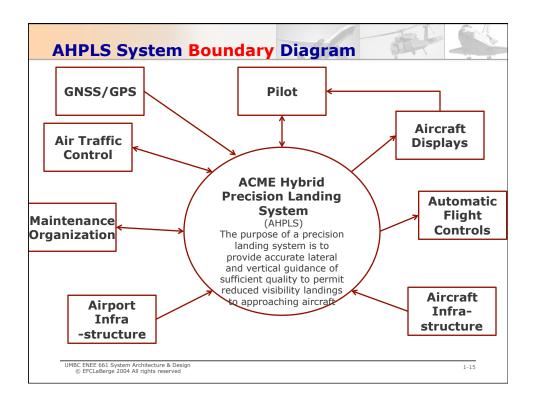
- Graphics assist in communication with customers, coworkers and other stakeholders
- Text documents will flow from the artifacts.
- Heuristic: A picture really is worth 1000 words!

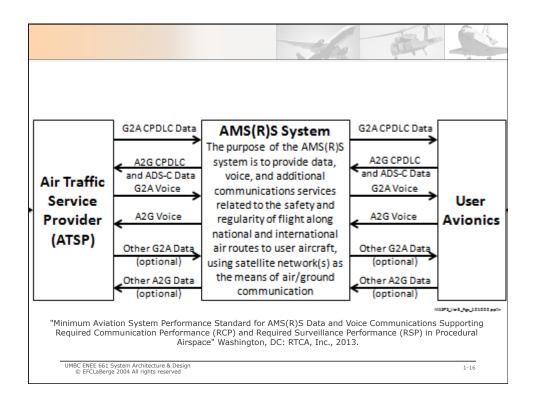
UMBC ENEE 661 System Architecture & Design
© EFCLaBerge 2004 All rights reserved

System Boundary Diagram (SV-1)

- Defines the context/scope of the work to be provided as part of your product or system.
- Identifies the external adjacent systems
- Heuristic: Diagram abstraction should result in 7±2 elements.

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved





What do we know about the interfaces?

- Tool is the Information Exchange Matrix (OV-3)
- Place to capture tribal knowledge
- Very important to keep a high level of abstraction
- Identify problematic interfaces
 - Interfaces for which we don't know what, how, how much or how fast
- Don't focus on implementation details!

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved

Example attributes for information exchange matrix...

- Source
- Destination
- ID
- Description of the information received/ transmitted
- Media for the interface (wire/wireless/network, etc)
- Analog/Digital
- Size/Volume
- Type (Autonomous/Cooperative/Active)
- Update Rate
- Bandwidth
- Security
- Safety Criticality
- Mission Criticality
- Error Rate

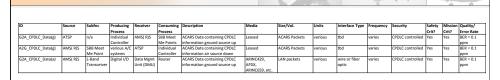
UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved 1-18

Sample Information Exchange Matrix

- Write down everything you know at this level.
- Keep the level of abstraction appropriate
- Heuristic: It doesn't exist if it isn't written down!

	Information Source			Inf. Des	stination										
ID	OIE Source	Sub- Function	Producing Activity	OIE Receiver	Consuming Activity	Description	Media	Size/Vol.		Interface Type	Freq.	Security	Safety Critical		Quality/F OM/Error Rate
	i														

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved



ID	Source	Subfnc	Producing	Receiver	Consuming	Description	Me
			Process		Process		
G2A_CPDLC_Data(g)	ATSP	n/a	Individual	AMS(R)S	SBB Meet	ACARS Data containing CPDLC	Leas
			Controller		Me Points	information ground source up	
A2G_CPDLC_Data(g)	AMS(R)S	SBB Meet	various A/C	ATSP	Individual	ACARS Data containing CPDLC	Leas
		Me Point	systems		Controller	information air source down	
G2A_CPDLC_Data(a)	AMS(R)S	L-Band	Digital I/O	Data Mgmt	Router	ACARS Data containing CPDLC	ARII
		Transceiver		Unit (DMU)		information ground source up	AFD

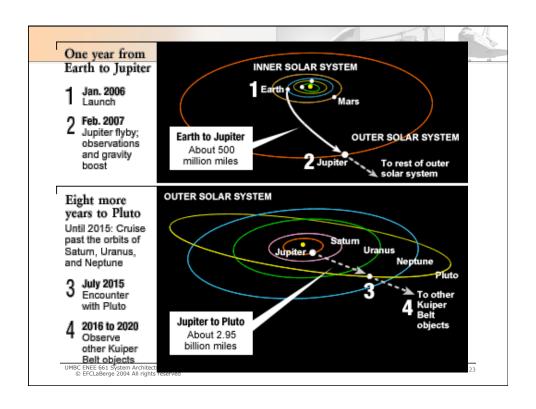
Media	Size/Vol.	Units Interface Typ		Frequency Security		Safety	Mission Quality/	
						Crit?	Crit?	Error Rate
Leased	ACARS Packets	various	tbd	varies	CPDLC controlled	Yes	Yes	BER = 0.1
								ppm
Leased	ACARS Packets	various	tbd	varies	CPDLC controlled	Yes	Yes	BER = 0.1
								ppm
ARINC429,	LAN packets	various	wire or fiber	varies	CPDLC controlled	Yes	Yes	BER = 0.1
AFDX,			optic					ppm
ARINC659, etc.								

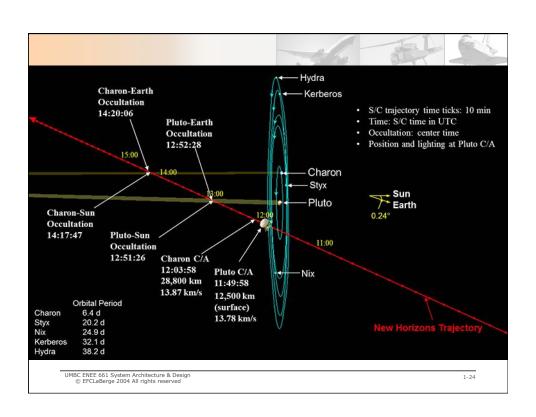
UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved 1-20

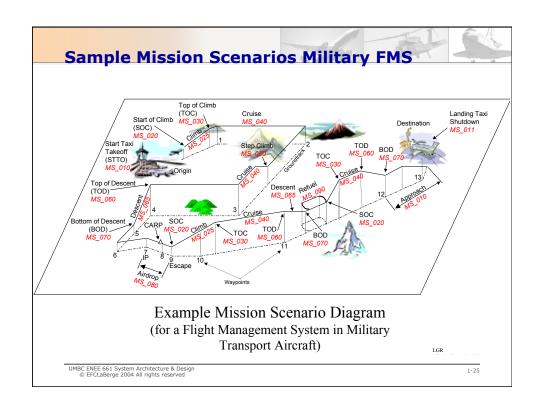
How is the system to be used?

- Sketch out a (series) of "mission" alternatives
 - Drawing doesn't need to be complicated
- Identify the key events and when they occur
- Identify what the consumer actions are
 - E.g. push a button, turn power on, type command on keypad, etc.
- Identify external responses from the system
 - What changes can the user perceive?

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved



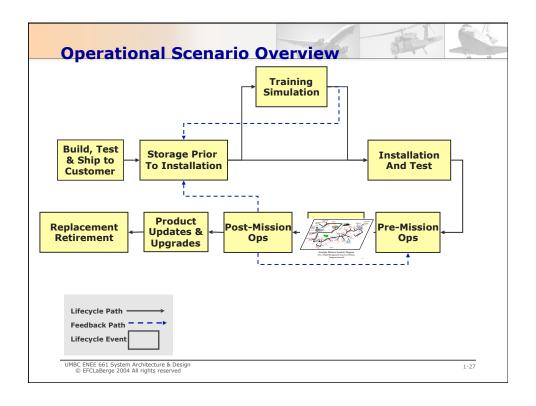




The operational scenarios

- Consist of life-cycle events
 - Manufacture
 - Storage
 - Training
 - Installation
 - Mission Operation (previous graphic)
 - Post-mission
 - Maintenance
 - Upgrades
 - Replacement or retirement

UMBC ENEE 661 System Architecture & Design © EFCLaBerge 2004 All rights reserved



Summary

- The importance of the operational view
 - Help clarify user demands
 - Consider all 4 C's
 - Customer
 - Consumer
 - Client
 - Caretaker
- What's in the system? (SBD)
- What are the interfaces? (IEM)
- How will we use it? (MSD)
- How will it grow or age? (OSD)
- Next time: The Functional View

 UMBC ENEE 661 System Architecture & Design
 © EFCLaBerge 2004 All rights reserved