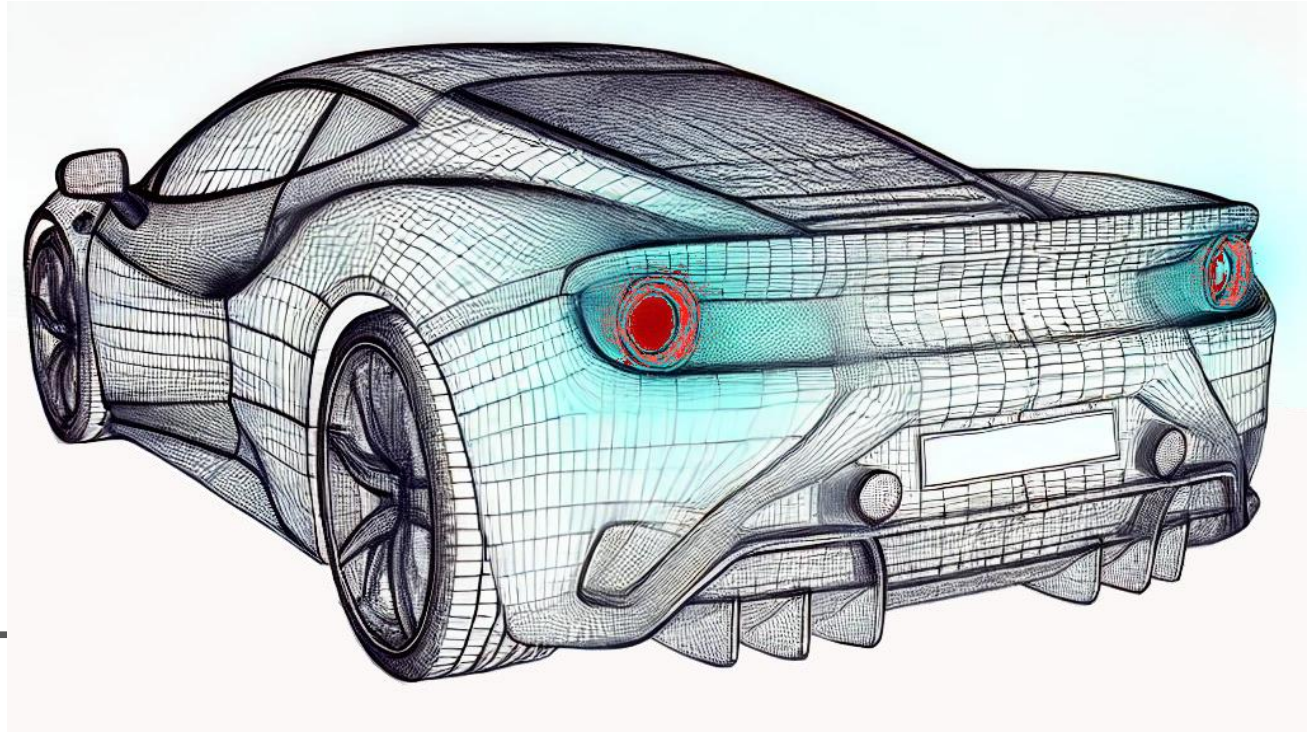
SOLUTIONS FOR COMPLEX SYSTEMS



Systems Integration

A Complex Challenge

David Hetherington
SYSTEM STRATEGY, INC.
Principal
dhetherington@systemxi.com

2024-12-13

Agenda

Agenda

- Personal Introduction
- Overview of Systems Integration
- Economics and Sociology
- Common Antipatterns
- Case 1 : The Jackup Rig
- Case 2 : The Infotainment System
- Success in Type A Situations
- Success in Type B Situations
- Questions?

Personal Introduction

Leaving Pearl Harbor, May 2018

Personal Introduction

May 1, 2018. David Hetherington with US Navy nuclear officer Ryan Hetherington on the deck of the USS Theodore Roosevelt (CVN-71) pulling out of Pearl Harbor during a “Tiger Cruise” to San Diego.

Multiple Roles

Personal Introduction

ヘサリントン友子

David Hetherington

System Strategy, Inc
dhetherington@systemxi.com
Boutique MBSE Consultancy
~20 people
Defense/Commercial

Asatte Press, Inc
David.Hetherington@asattepress.com
Publishing
2 people / 5 authors

Industry Groups
David_Hetherington@ieee.org
Standards, Industry Working Groups

Odyssey of Homer Career

Personal Introduction

Digital Phone Systems

High Availability

Real-Time Control

Multimedia Audio and Image

Digital Signal Processing

Laptop Design

Booming businesses
That turned into pumpkins

Enterprise Systems Management SW

Software Internationalization

CORBA

Storage

Networking

Data Center HW

Blade Servers

Challenging projects

Engineering Services

Services Businesses

Automotive Radar

Radar Math

Custom Chipset

MBSE

Automotive Business

Digital Publishing Toolchains

MBSE

XML toolchains

Offshore Oil Drill Ships

SW Safety

MBSE

Automotive Infotainment

Automotive Design

Automotive Semiconductors

ISO 26262

MBSE

MBSE Consulting

MBSE

Overview of Systems Integration

When Stuff Comes Together

Overview of Systems Integration

This process is challenging when everything is sourced internally.

When the suppliers are all separate companies, the process becomes extremely difficult.

Supplier 2

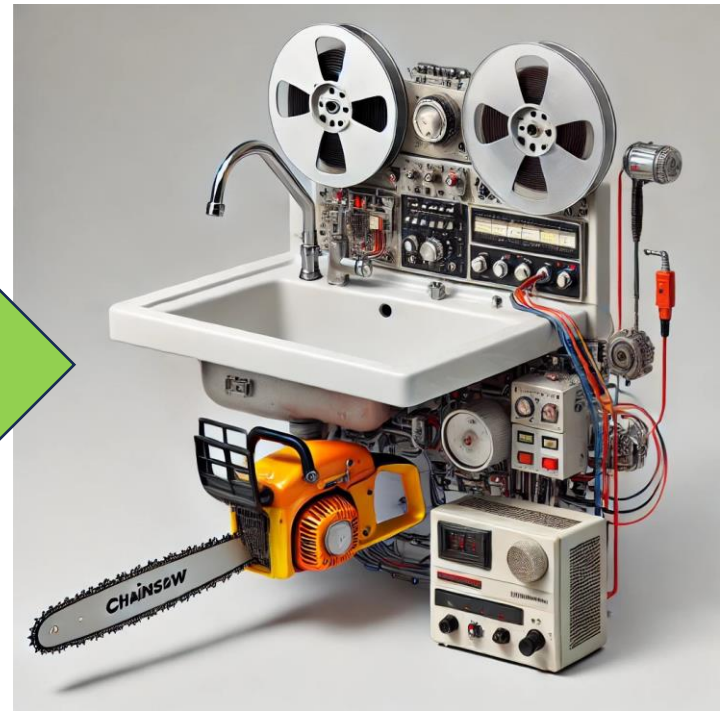
Supplier 1

Supplier 3

Supplier 4

Supplier 5

Systems Integration



Finished System

Two Common Organization Types

The Marching Band

The Goat Rodeo

Two Common Organization Types

The Marching Band

Funding Cycle	10 Years
Capability Maturity	High = “Optimizing”
Repeatable Projects	Dozens of cycles of similar projects
Team Stability	High
Management Focus	Fine tuning effectiveness

The Goat Rodeo

Funding Cycle	Erratic
Capability Maturity	Low = “Inital”
Repeatable Projects	Every project is unique
Team Stability	Constant turnover
Management Focus	Survival

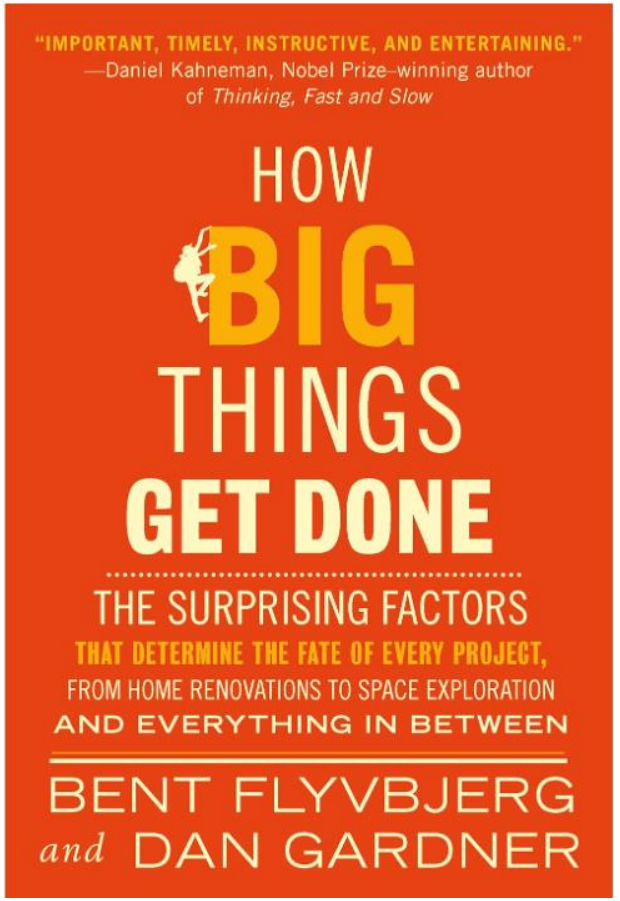
How Big Things Get Done

Overview of Systems Integration

Book Recommendation

Decades of Research, 16,000 Projects in Database

Why Some Succeed, But Most Fail



https://www.amazon.co.jp/-/en/Bent-Flyvbjerg-ebook/dp/B0B3HS4C98

Thinking/Acting Fast and Slow

Think Fast
(Bold, Decisive)

Act Slow
(Mired in Unexpected Problems)

Example: Sydney Opera House

Think Slow
(Measure Twice, Cut Once)

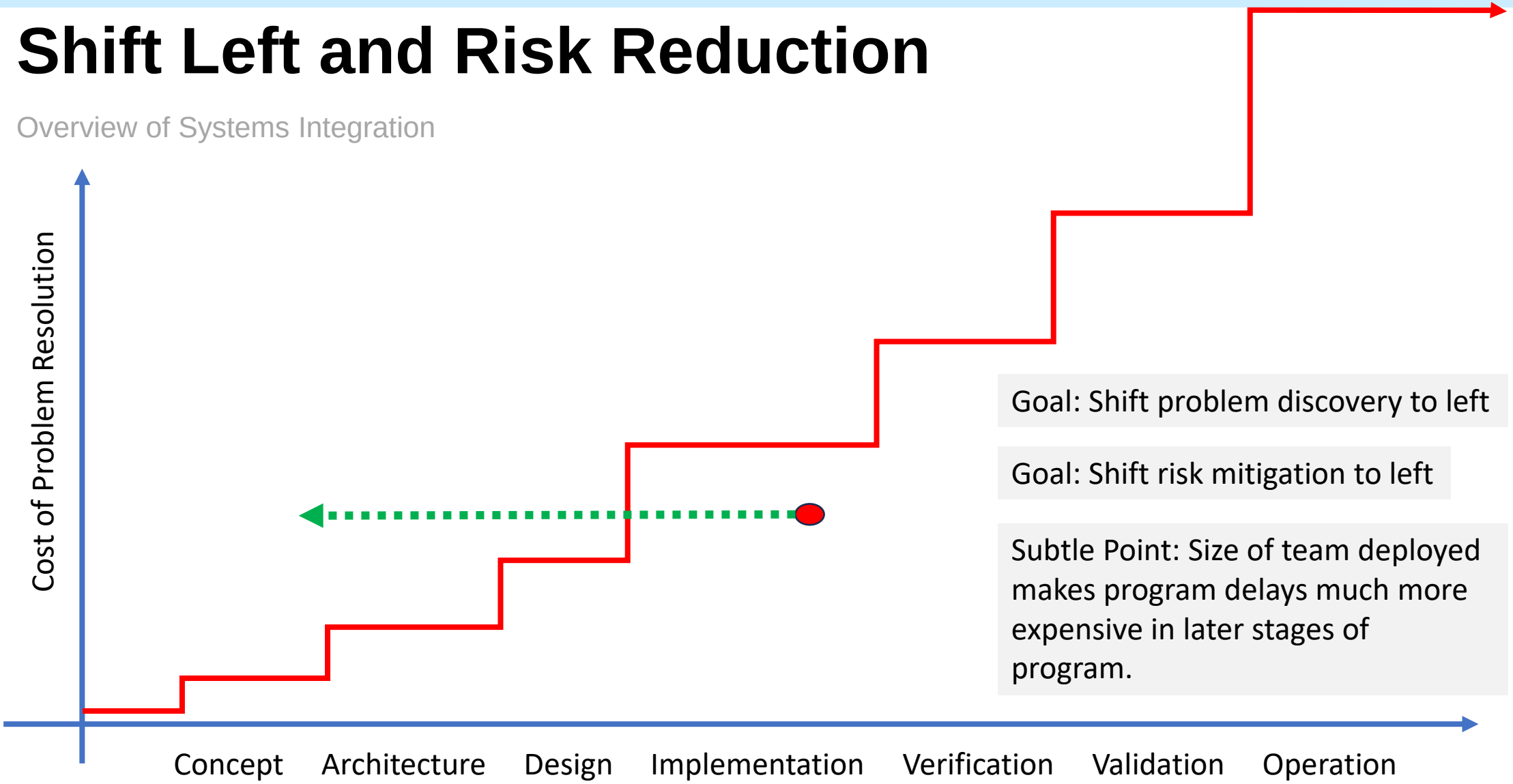
Act Fast
(Predictable Implementation)

Example: Frank Gehry

https://hbr.org/2023/01/how-frank-gehry-delivers-on-time-and-on-budget

Shift Left and Risk Reduction

Overview of Systems Integration

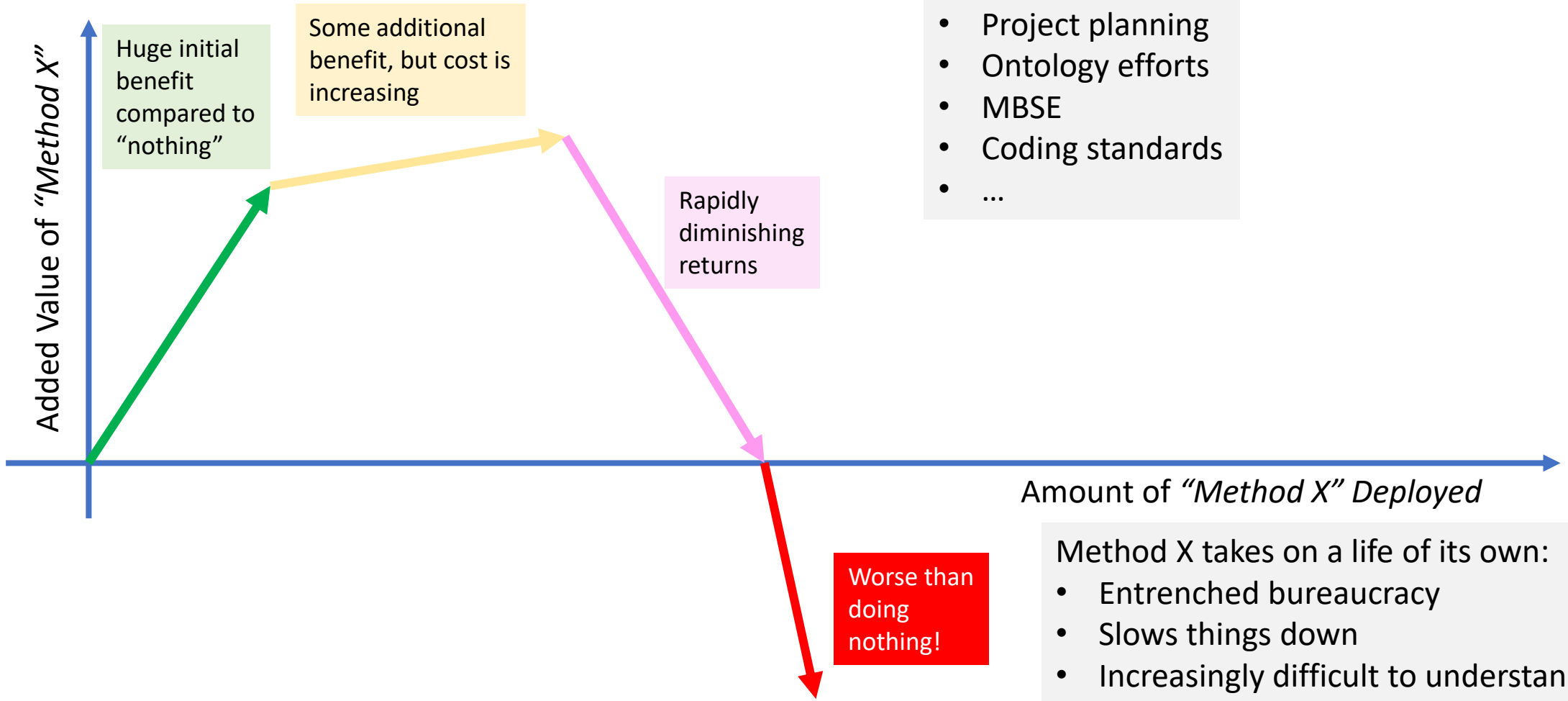


Goal: Shift problem discovery to left

Goal: Shift risk mitigation to left

Subtle Point: Size of team deployed makes program delays much more expensive in later stages of program.

The Hetherington Curve for Method X



Applies to:

- Project planning
- Ontology efforts
- MBSE
- Coding standards
- …

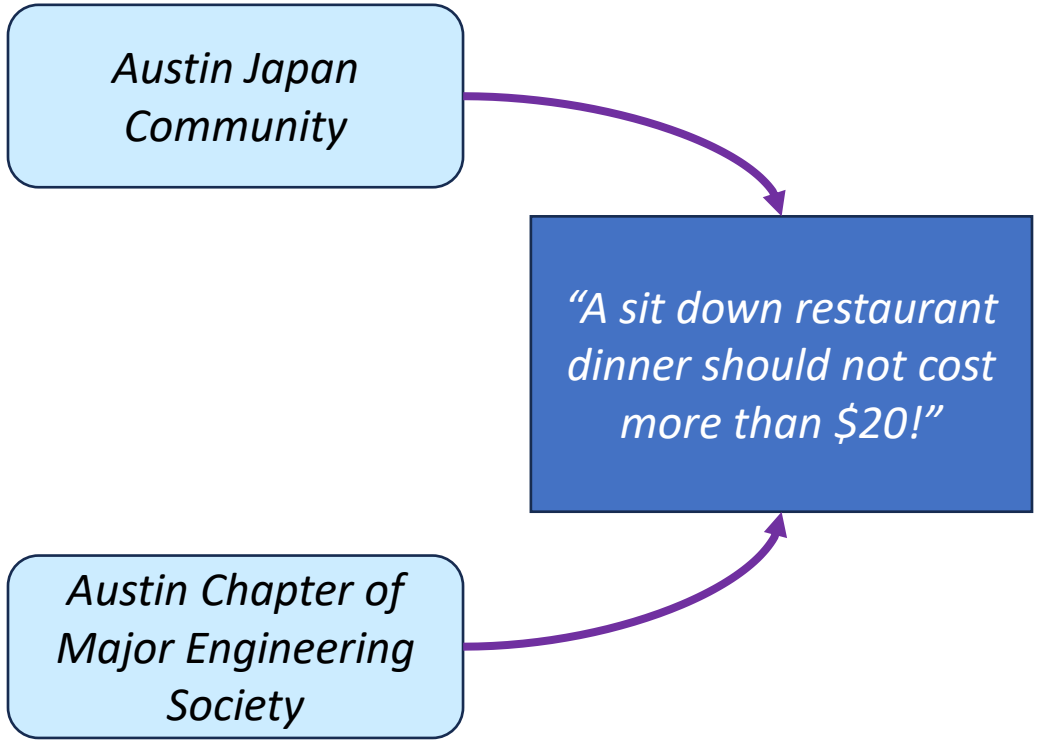
Method X takes on a life of its own:

- Entrenched bureaucracy
- Slows things down
- Increasingly difficult to understand
- Lower "real life" fidelity

Economics and Sociology

The Bias Towards Cost Underestimation

Economics and Sociology



Two completely unrelated populations in Austin.

Both populations had a rigid and unshakeable price expectation.

Meeting that expectation was difficult pre-COVID….

After COVID, it was completely impossible.

Reality post-COVID was in the $45-$50 range…

However, the always less-than-realistic price expectation did not budge in either population.

The Net Present Value Problem

Economics and Sociology

Year	1	2	3	4	5
Cash Out	$200	$400	$500	$600	$800
Cash In	$0	$0	$200	$1,000	$2,000
Net Cash	($200)	($400)	($300)	$400	$1,200
Required Return		8%	8%	8%	8%
Discount Factor	1.00	0.93	0.86	0.79	0.74
Discounted Cash Flow	($200)	($370)	($257)	$318	$882
Net Present Value	**$372**				

"If all cost factors and risks are considered carefully, it is extremely difficult to find projects with a positive net present value"
Professor Ramesh K.S. Rao
University of Texas McCombs School of Business

This project would look good to an investor.

For most projects, the expected positive cash flow is far in the future and gets heavily discounted…

Meanwhile initial costs do NOT get discounted much.

Net effect makes it difficult to budget "enough" cash or schedule time.

Strategic Misrepresentation

Economics and Sociology

These combined economic effects force a certain level of “Strategic Misrepresentation” to get almost any project off the ground.

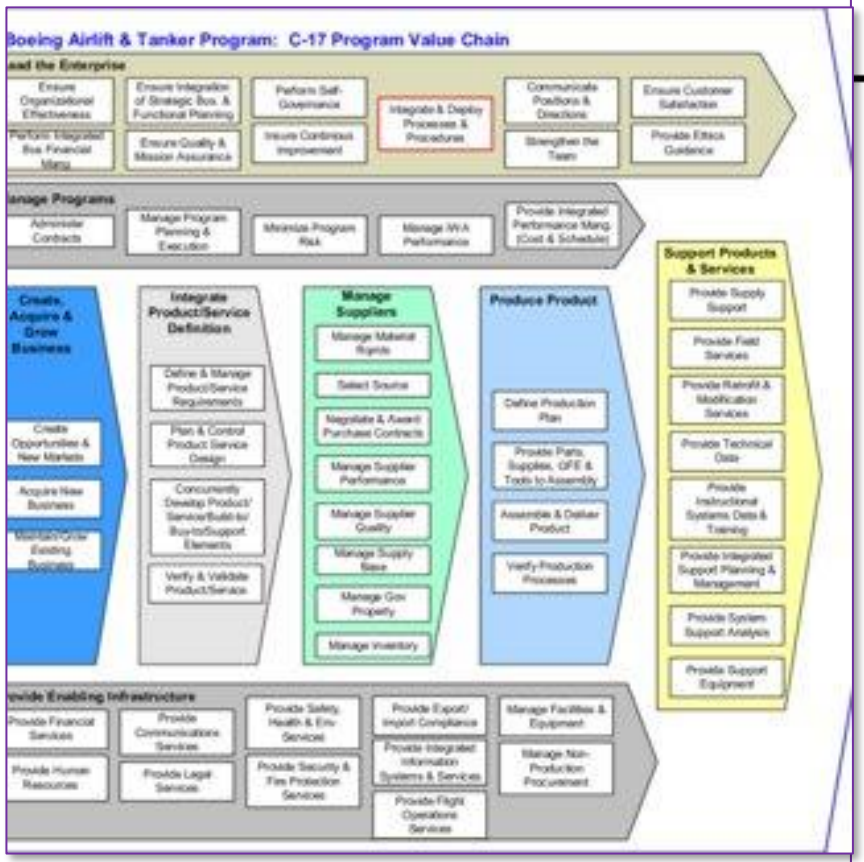
Most often, this involves strategically “forgetting” to include certain aspects of the cost…

Systems Integration is a perfect candidate to be wishfully underestimated or neglected entirely.

A related problem is the need to show “shovels in the ground” progress very rapidly to prevent the stakeholders from changing their minds and withdrawing the funding.

The Boeing/NASA Systems Integration Process

Economics and Sociology

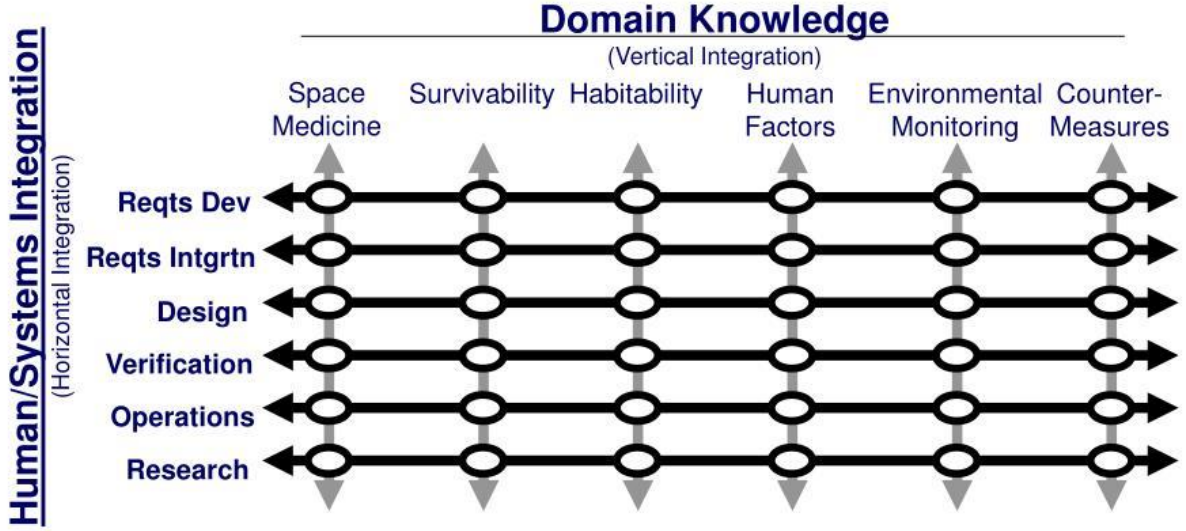


An Approach to Horizontal Integration

NASA HSI Experience in an SE Environment

This paper proposes that the Core Capabilities provide a means of breaking Horizontal Integration into manageable components

- Each component has its own constraints and procedures that guide cross-domain integration



8

The Space X Systems Integration Process

Economics and Sociology

To be clear, Space X ***does*** have systems engineering processes!

However, the Space X systems engineering processes are obviously calibrated for a laser focus on business results.

Exploding rockets are seen as necessary learning exercises to get to commercial viability rapidly…

…not as organizational humiliations to be avoided at all costs.

(Part of the difference is private funding versus tax payer funding)

Experiments and Technology Readiness Level

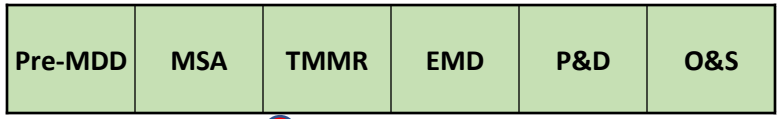
Economics and Sociology

Where is your organization on the TRL scale? If they are in the "business experiments" stage, you can hurt them by imposing a heavyweight Systems Engineering process.

Experiments

Here we are trying to narrow down to a ***single factor*** to test a hypothesis.

Technology Transition



SRR = "System Requirements Review"

Here we are trying to consider ***all possible factors*** to make sure the system will perform effectively and safely in the widest possible variety of conditions.

Systems Engineering

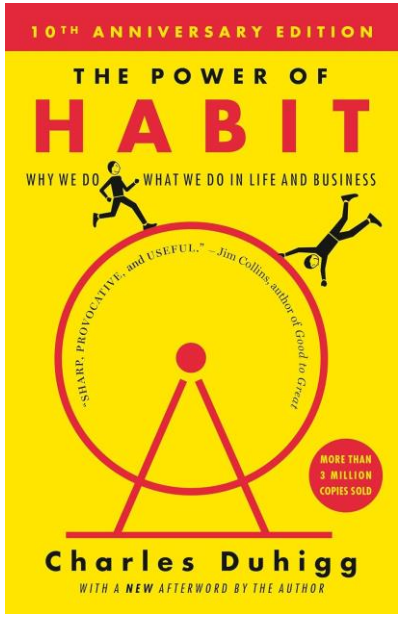
TRL : https://www.gao.gov/assets/gao-20-48g.pdf (January 2020)
Procurement Phases: Engineering of Defense Systems Guidebook (February 2022)

Trying to Teach Pigs to Sing

Economics and Sociology

Some organizations will never adapt to a careful, disciplined systems engineering approach, no matter how much effort you invest in trying to convince them.

Even worse, they will rapidly group you together with that humorless, nagging, elementary school teacher that they hated.



Recommended reading!!
Read the chapter on Paul O'Neil rescuing Alcoa by getting everyone focused on safety. This chapter is fascinating and reading it will brighten up your day and give you some positive hope and ideas on how to turn around a difficult cultural situation.

https://www.amazon.com/Power-Habit-What-Life-Business/dp/081298160X

"Never try to teach a pig to sing.
It wastes your time…
…and it annoys the pig."

Common Antipatterns

Enjoying the Shopping Trip

Common Antipatterns

Shopping for shiny new toys is very enjoyable.

Sweating the messy details of getting all the shiny toys working together is much less enjoyable.

The mess lands in the lap of the systems integration team.

Affects both of the Cases.

The Late Dinner Guest

Common Antipatterns

The dinner party hostess has crafted a detailed plan:

- *Dress: White Tie*
- *6:00 – 6:20 – Cocktails on the terrace*
- *6:20 – 6:40 – Personal flambeed hors d'oeuvres*
- *6:40 – 6:50 – ….*

Unfortunately, one inconsiderate guest arrives very late, inebriated, inappropriately dressed, and dragging along an uninvited additional guest…

This is a common problem with meticulously planned, hierarchical systems integration projects.

At least one of the subsystems will show up late, unstable, displaying unplanned behaviors, and dragging along unannounced extra necessary adapter boxes.

Pushing the Problem onto Someone Else

Common Antipatterns

Unable or unwilling to deal with the systems integration problem, sometimes system owners will attempt to push the burden off onto some other party.

If the other party is a professional systems integration shop and is properly compensated and supported by the system owner, this sort of approach can work very well.

Unfortunately, the designated party often lacks the systems integration know-how, does not want the job, and is not properly compensated or supported by the system owner.

Affects both of the Cases.

Procrastination in Resourcing

Common Antipatterns

Organizational Amnesia

Common Antipatterns

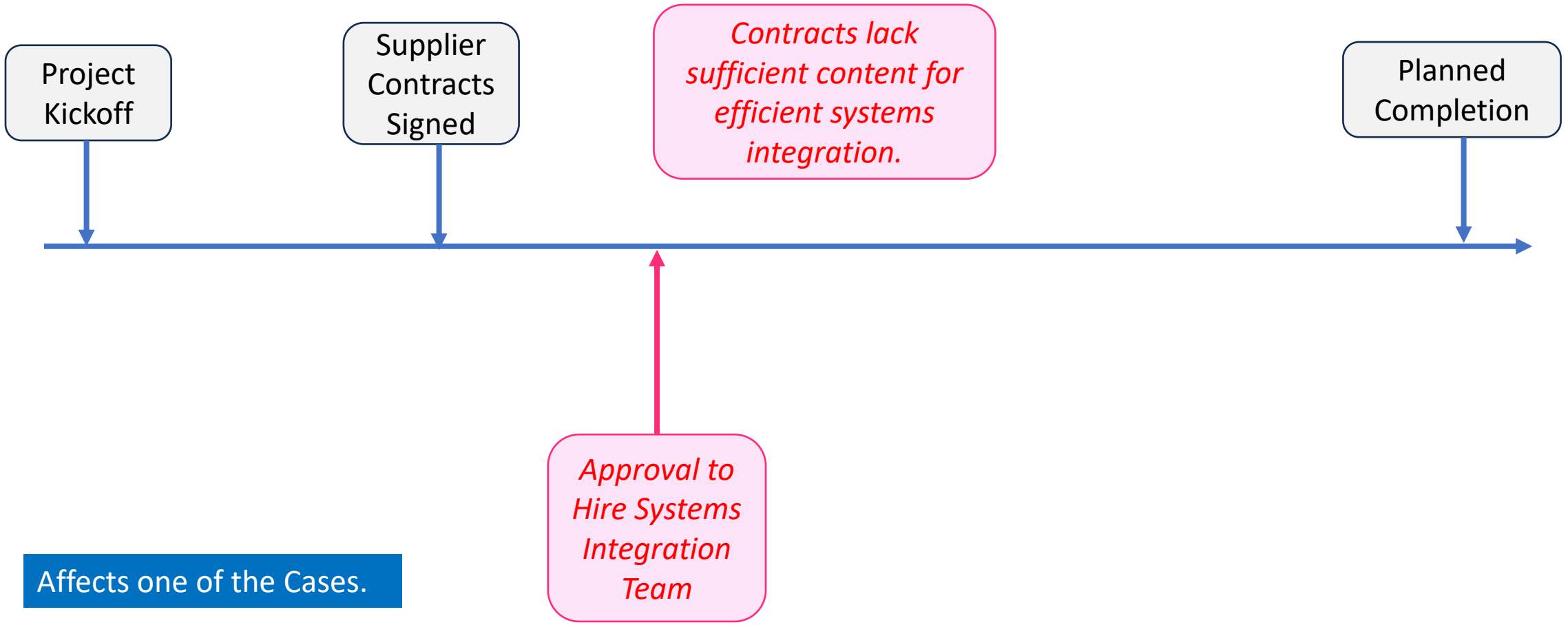
Organizations display a remarkable capacity to forget problems in previous projects and repeat the same mistakes.

Even if the problems are remembered, they are often discounted as “bad luck”.

In the euphoria of kicking off a new project, there is a tremendous tendency to declare that this time: *“All obstacles will be overcome by agile, proactive, teamwork, and can-do spirit.”*

Missing Contractual Terms

Common Antipatterns



Affects one of the Cases.

The All-You-Can-Eat Risk Buffet

Common Antipatterns

The organization walks down the “All-You-Can-Eat Risk Buffet” and piles every conceivable risk onto the project.

Several different sociology effects contribute to this problem.

Affects one of the Cases.

Case 1 : The Jackup Rig

https://en.wikipedia.org/wiki/Jackup_rig

Project Overview

Case 1 : The Jackup Rig

A Singapore Shipyard was building a “jackup rig” – a specialized oil drilling ship.

The owner was a major drilling fleet operator.

The initial main customer was a European oil company.

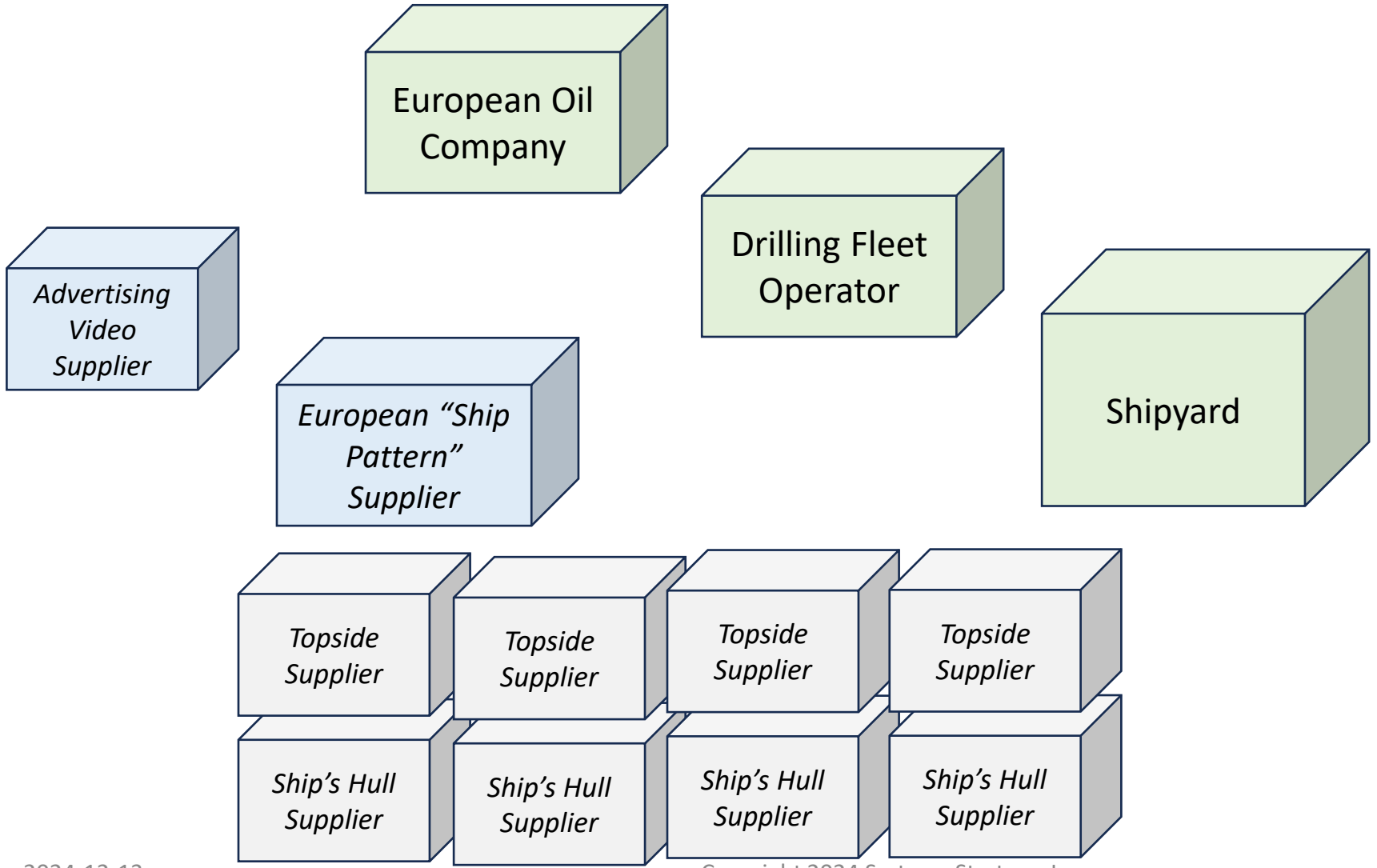
In the wake of the Deep Water Horizon disaster, the oil company demanded certification to the DNV “ISDS” software safety standard. (DNV later merged with GL to form DNVGL.)

Since the shipyard had no such expertise, they contracted consulting from a small, Texas-based, specialty engineering company.

I was soon flying back-and-forth, 28-days on, 28-days off.

The Stakeholders

Case 1 : The Jackup Rig



N! communication paths.

All three "green" having some level of parallel communications with all "blue" and "grey" parties.

No written system specification – only the contract.

Contract is a pile of previous contracts, e-mails, etc.. – all spiral bound together, but not otherwise coherent.

Overview of the Jackup Rig

Case 1 : The Jackup Rig

- Specialized oil drilling ship
- All features of ship except propulsion
  - Quarters for 150 crew
  - Galleys, laundry, etc..
  - Lifeboats
  - Anticollision radar
- Three 200 meter legs
  - Rammed 2 meters into ocean floor
  - After that, ship lifts >60 meters above ocean
  - 5000 ton drilling derrick skidded over edge
  - Drilling crew works in the suspended derrick

https://en.wikipedia.org/wiki/Jackup_rig

Systems, Sensors, and Actuators

Case 1 : The Jackup Rig

- Hollow oil drill bit invented by the father of Howard Hughes
- Slurry of mud forced down the center of the drill pipe by high-power pumps (Megawatts of power)
- Slurry flows out holes in the drill bit and back up the well.
- Slurry cools the drill bit.
- Slurry carries up:
  - Oil
  - Water (contaminated)
  - Rock chips
  - Poisonous gas
  - Explosive gas
- Mud pressure is crucial for safety. The master driller has everyone's lives in his hand.

Ship electronics included:

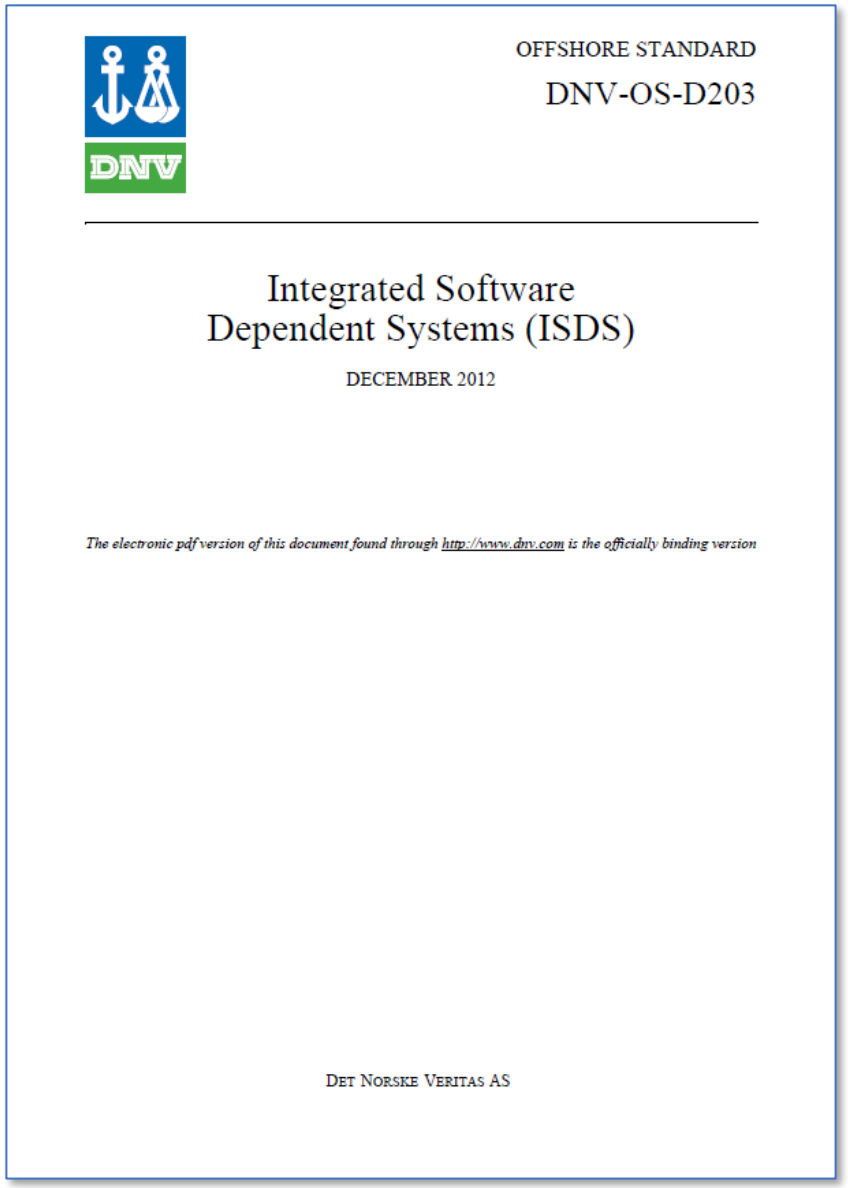
- 100 server computers
- 10,000 sensors
- 3,000 programmable logic controllers

DNVGL's ISDS

Case 1 : The Jackup Rig

- DNV = “DET NORSKE VERITAS” was one of the historical ship classing society along with Lloyds of London, American Bureau of Shipping, Germanischer Lloyds, Bureau Veritas, and Class NK in Japan.
- Original function was similar to financial audit companies. Inspect ships on behalf of insurance companies.
- By 2013, several of these classing societies had developed inspection standards for software safety.
- These were similar to ISO 26262 for the auto industry.
- Basic inspection was to demonstrate disciplined control of requirements, verification, traceability, and so on.

- Almost no shipyard in the world had this sort of careful culture as of 2013.
- Coaching a shipyard into compliance was painful.



OFFSHORE STANDARD

DNV-OS-D203

Integrated Software Dependent Systems (ISDS)

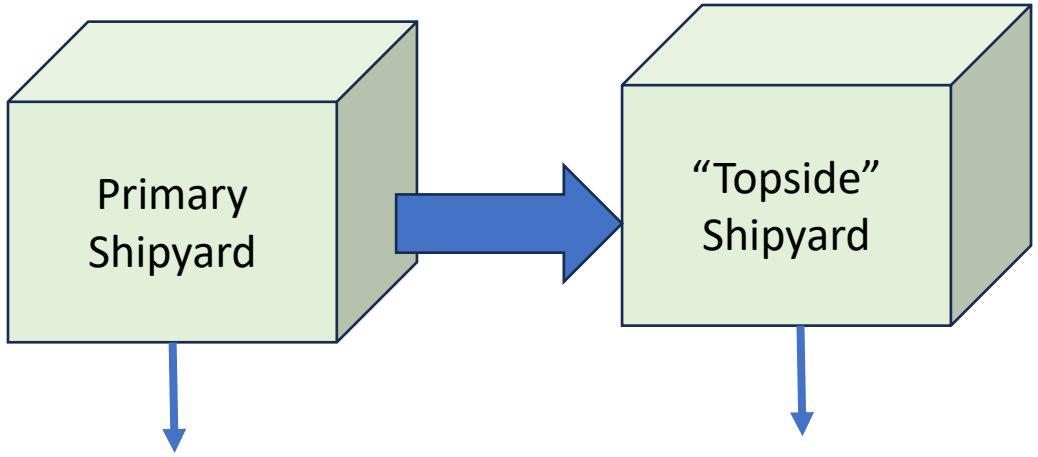
DECEMBER 2012

The electronic pdf version of this document found through http://www.dnv.com is the officially binding version

DET NORSKE VERITAS AS

Shipyard Specialization

Case 1 : The Jackup Rig



Basic ship:

- Hull
- Propulsion
- Steering
- Living quarters
- Life boats

Integrate specialized "topside" equipment

- Drilling systems
- Processing systems
- Helicopter operations

Standard industry practice around 2013 was to use two different shipyards to create a specialized drillship.

The primary shipyard would create "The Ship".

The second shipyard would integrate the specialized equipment.

In this case, the European oil company and drilling fleet operator had forced the Singapore shipyard to cover both roles.

The Singapore shipyard was not particularly happy with being forced into this role.

140+ Standards, 50 Years, 30 Jurisdictions

Case 1 : The Jackup Rig

The first item of business was to nail down “The Software Requirements”.

The spiral bound conglomeration of stuff did not include any section identified as “software requirements”. It did, however, refer to a large number of standards.

In fact there were more than 140 different referenced standards, written by more than 30 different jurisdictions and authorities over a span of about 50 years.

None of these standards contained the word “software”. However, they did contain phrases like *“….the door shall close automatically.”*

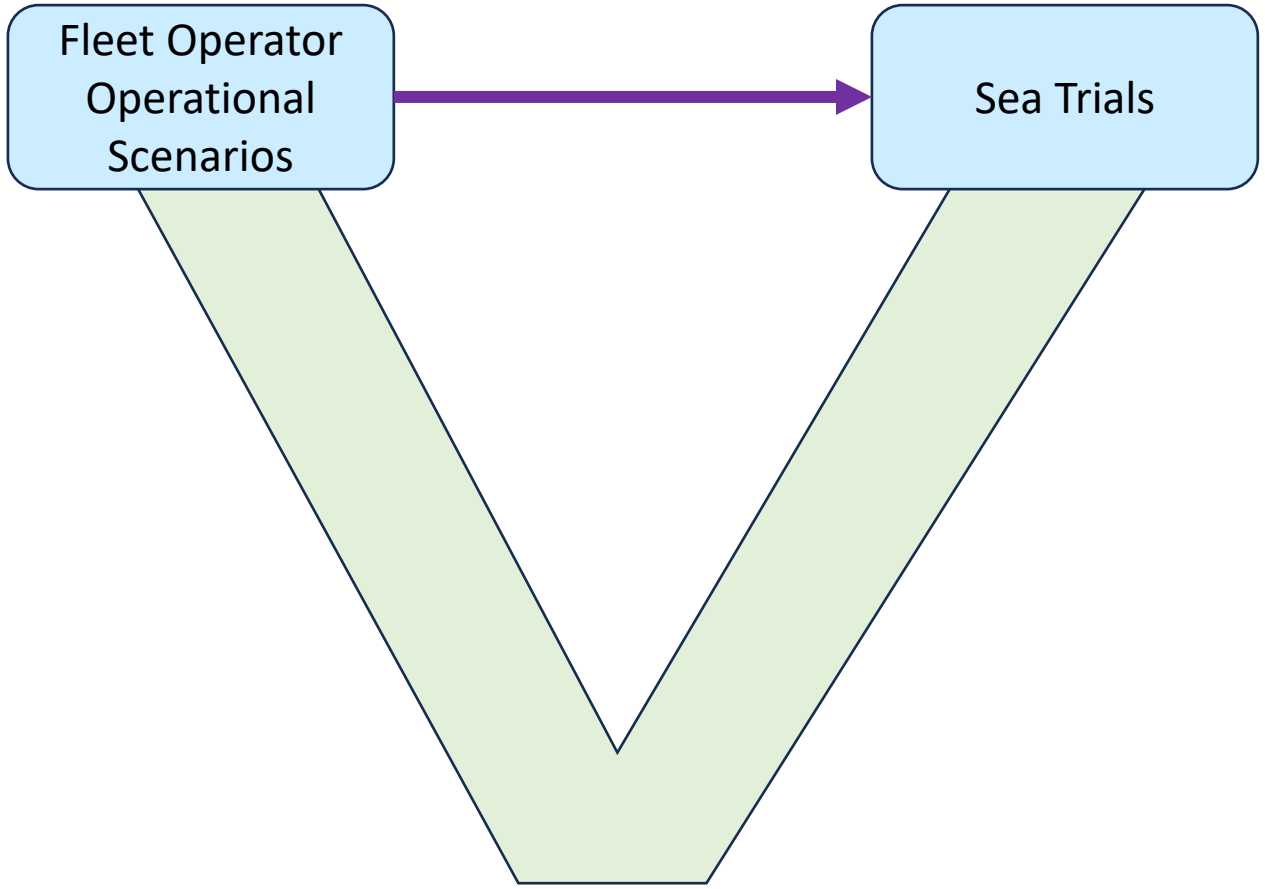
The standards did contain a large number of tangled cross references to each other.

Strategy: Create a SysML model that was a library of packages for each standard containing SysML requirement elements for each blob of suspicious text.

“Dave, why do we need to write these requirements down? Everyone knows how these things work already!”
Young mechanical engineer serving as the shipyard’s project leader.

Validation, Sea Trials, and Commissioning

Case 1 : The Jackup Rig



Many organizations get very confused about the difference between “verification” and “validation”

The ship building industry actually has had a much better term for “Validation” and that is “Sea Trials”

In support of this, the drilling fleet operator had provided a very nice set of easy-to-understand “operational scenarios”.

Very nice.

Systems Integration: Phase 2

Case 1 : The Jackup Rig

Unfortunately, in many cases “Sea Trials” would fail miserably…

Which would automatically trigger *“Systems Integration: Phase 2”*

The Jackup rig would be loaded onto a heavy lift ship for delivery to the first deployment area.

Onboard the Jackup rig (which would be riding on the heavy lift ship) would be 100+ specialized technicians, furiously tinkering and debugging trying to get the thing working.

Each of the specialized technicians would typically be billing $3000/day.

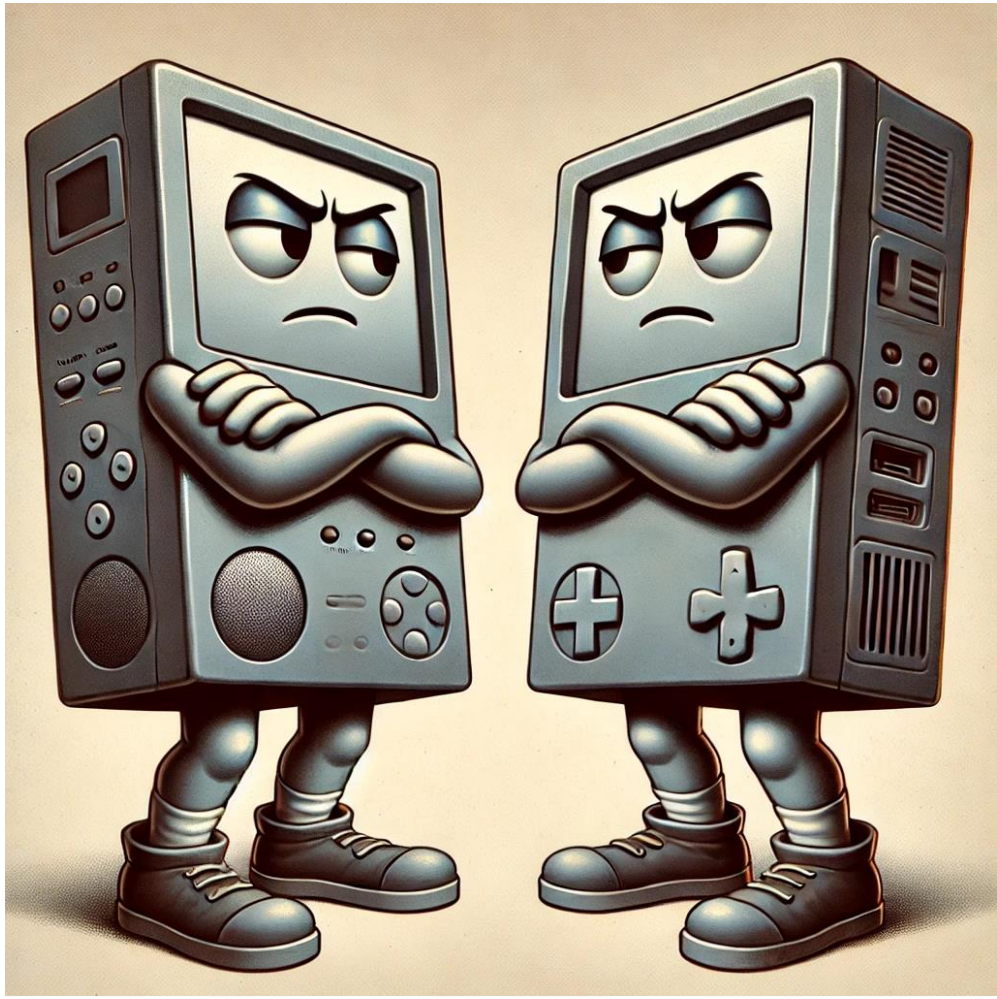
The oil industry seemed to regard this $300k/day cost overrun as a “routine cost of doing business”.

The Biggest Commissioning Pain Point

Case 1 : The Jackup Rig

I asked the shipyard project leader to arrange a meeting with the commissioning manager to talk about recurrent problems encountered during commissioning. (Commissioning being a shipyard term for systems integration)

He responded immediately and decisively: *”The biggest single problem is getting the boxes to talk with each other at all.”*



Showing the Suppliers my SysML Model

Case 1 : The Jackup Rig

OSI model

Layer			Protocol data unit (PDU)	Function[26]
Host layers	7	Application	Data	High-level protocols such as for resource sharing or remote file access, e.g. HTTP.
	6	Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption
	5	Session		Managing communication sessions, i.e., continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4	Transport	Segment, Datagram	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing
Media layers	3	Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control
	2	Data link	Frame	Transmission of data frames between two nodes connected by a physical layer
	1	Physical	Bit, Symbol	Transmission and reception of raw bit streams over a physical medium

https://en.wikipedia.org/wiki/OSI_model

Many of the PLC technologies in use in the ship were still working with older RS-232 and similar technologies.

Indeed, these have many DIP switches and manual protocol configurations.

DNV arranged a meeting of the key European equipment supplier engineers.

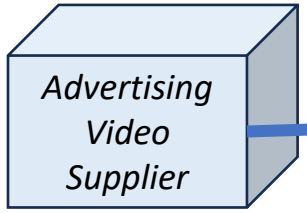
I reviewed the OSI model with them. I also showed them my (simple!) SysML model…

When I turned around, the entire room was staring at me wide-eyed as though I were a space alien crawling out of a flying saucer.

The Cost Overrun

Case 1 : The Jackup Rig

As part of the system concept activity, an advertising video company had been commissioned to create a beautiful simulation of a helicopter flying around the completed ship with all of the special features requested by the oil company.



Beautiful advertising video showing all of the changes requested by the oil company integrated into the standard ship pattern.

Unfortunately, the video company was staffed by graphic artists, not be mechanical engineers and they overlooked a special required conveyor belt required to properly remediate the rock chips coming out of the well. **Cost overrun ~ $1m**

By this time, the price of oil had fallen from a peak around $140/barrel to a low in the $20/barrel range. The oil company admitted that the oversight was their fault, but asked the shipyard to “work with them” to contain the cost.

To which the shipyard replied: ***“No problem at all! If we can just get rid of this useless software safety nonsense...!”***

Take Aways

Case 1 : The Jackup Rig

1 – Involuntary Systems Integration. The shipyard clearly did not want the job.

2 – Teaching Pigs to Sing. The supplier automation engineers were used to “tinker toy” ladder programming, did NOT have deep computer science or systems engineering skills, and really were not interested in new abstract methods. Shipyard engineers were similar.

3 – Functional Safety. Stakeholders clearly viewed software functional safety as a “nice to have” to be supported only if no inconvenience would be involved.

Case 2 : The Infotainment System

Project Overview

Case 2 : The Infotainment System

- One of the top minivans in the U.S. market
- New strategy by the OEM to own its own infotainment software based on Android
- Tier-1 suppliers to provide only hardware and driver support
- Major development center in rural mid-West location. Very difficult to staff top class software engineers.
- Major software development services organization tasked with 98% of the development effort
- Most of development team in Romania and Argentina
- Small team doing test and integration at OEM’s development center

The Auto Maker's Pain Point

Case 2 : The Infotainment System

A Tier-1 would typically charge $80m NRE for one Infotainment system for one geography.

The next year, the OEM would want one additional flashing lamp…

To which the Tier-1 would reply: *“Oh, that is going to be EXPENSIVE!”*

The OEM hated being vulnerable like this.

The Auto Maker's Game Plan

Case 2 : The Infotainment System

License a copy of Android from Google

Create distributed in-vehicle network of 9 different boxes for different function.

Separate contracts with multiple Tier-1s to provide the 9 different boxes.

More contracts with multiple specialty software suppliers.

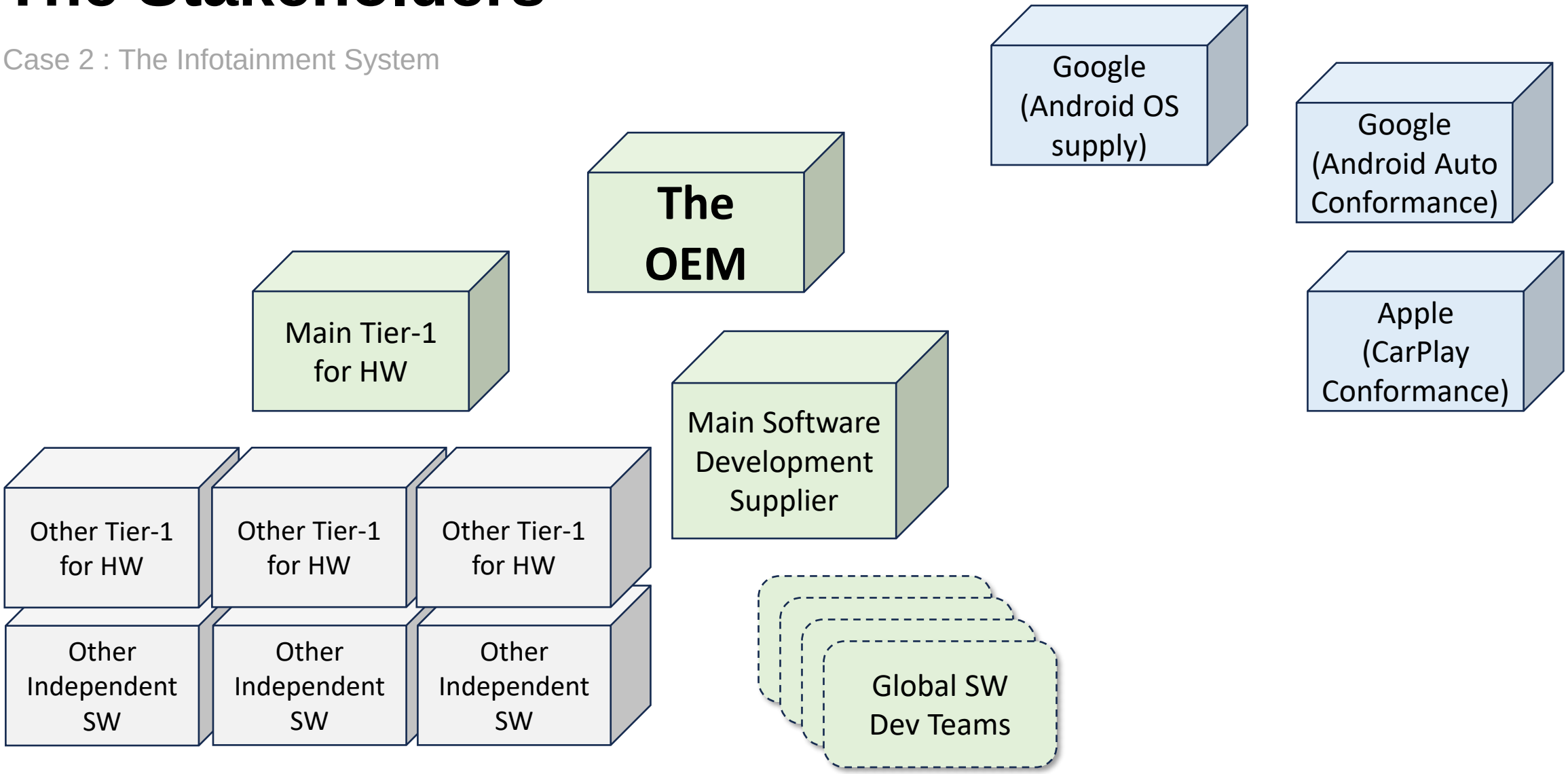
One large contract with the major software development supplier.

One key Tier-1 tasked with making the “main box” but with only the prospect of hardware sales.

The OEM could not understand why that Tier-1 was rather surly and uncooperative initially.

The Stakeholders

Case 2 : The Infotainment System



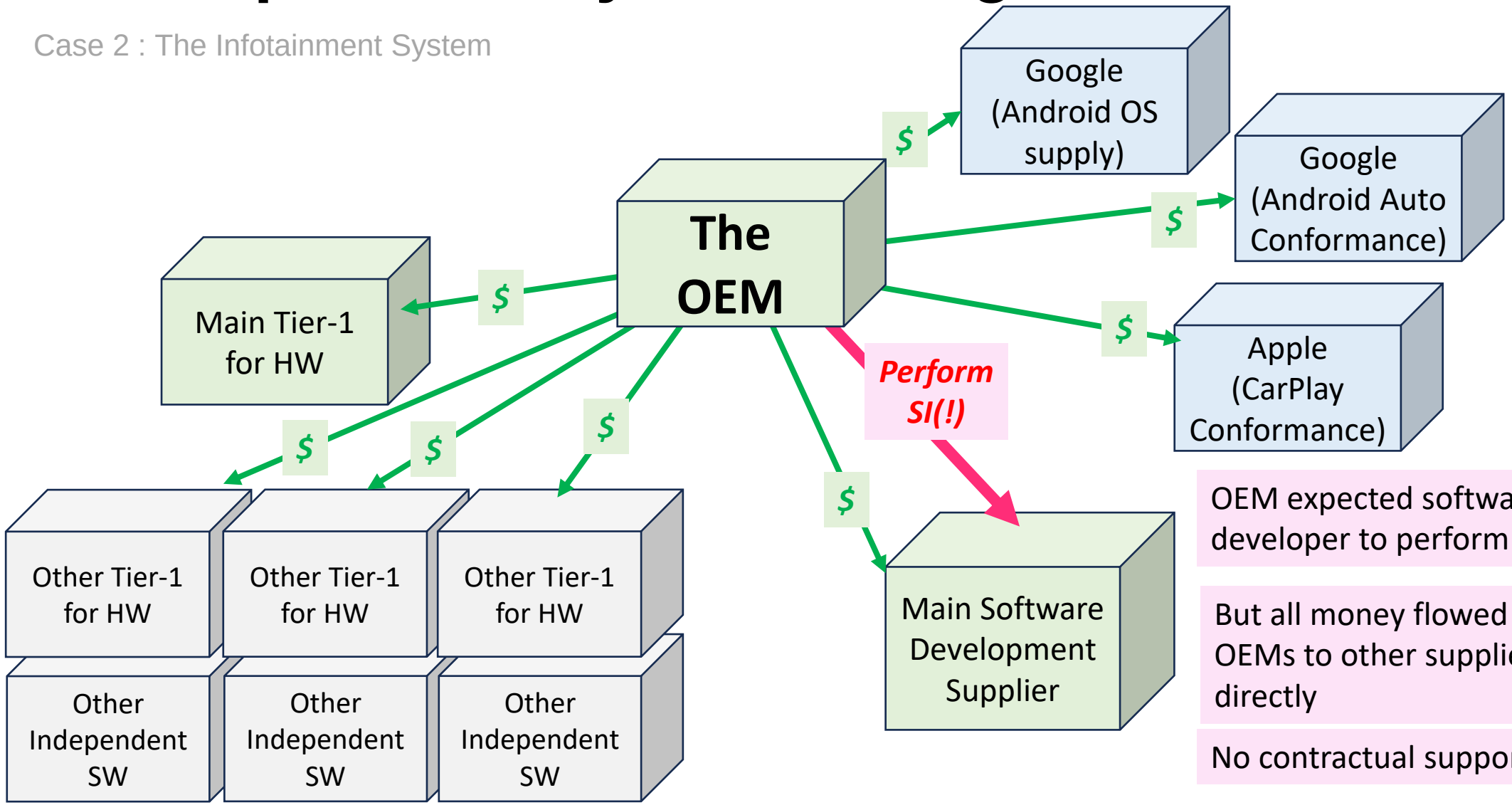
Visiting the All-You-Can-Eat Risk Buffet

Case 2 : The Infotainment System

- Overly Complicated Distributed Network
- Untested Networking Chip Technology
- Untested Networking Cable/Connector Technology
- Almost No Written Requirements
- Difficult Software Development at Remote, Rural Site
- Far Too Many Features
- Open Source Software Risks
- Refused Software Development Team's Attempts to Define Use Cases and Perform Formal Architecture
- New Supplier Business Model

The Impossible Systems Integration Role

Case 2 : The Infotainment System



OEM expected software developer to perform SI…

But all money flowed from OEMs to other suppliers directly

No contractual support at all

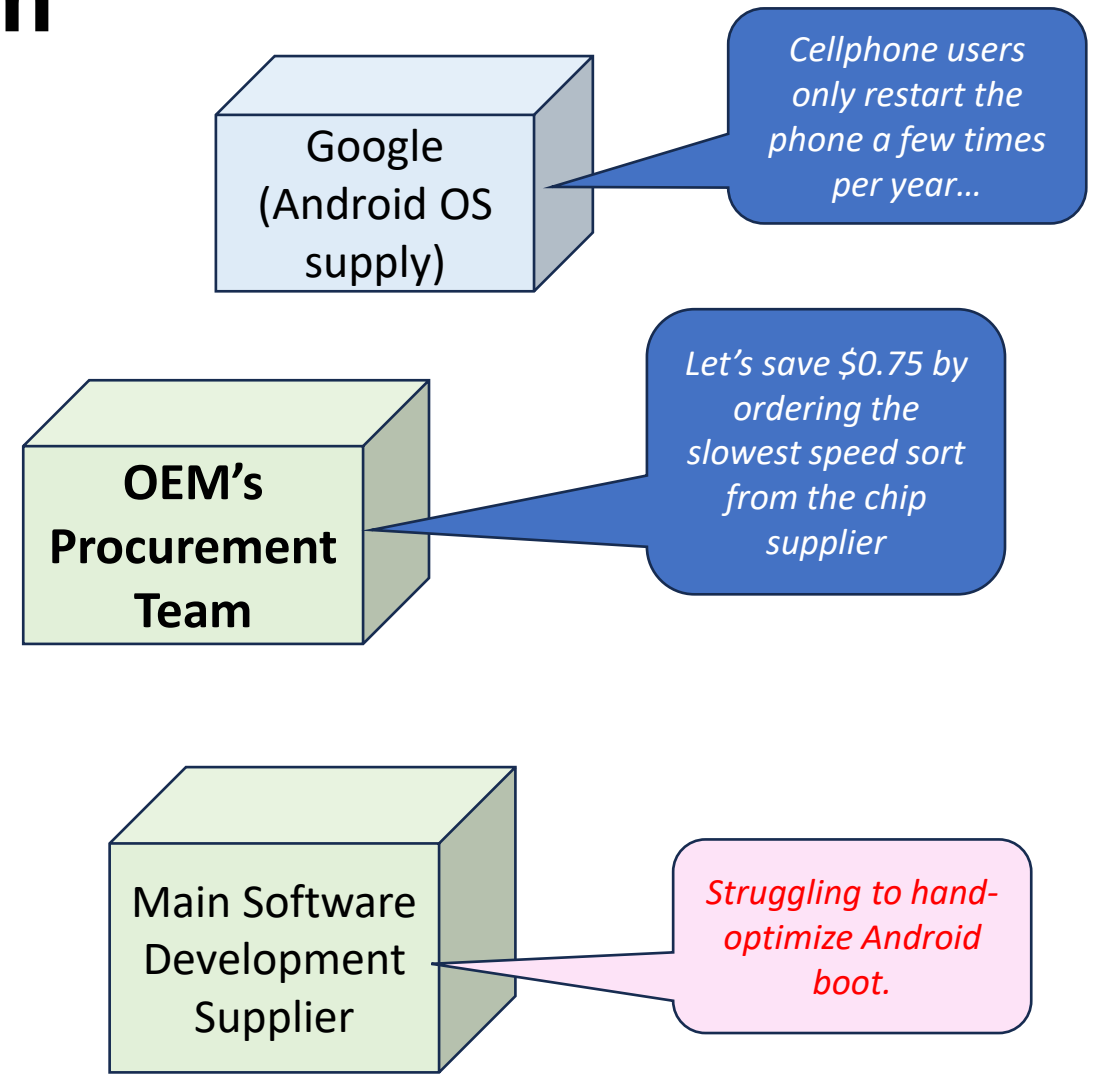
The Boot Speed Problem

Case 2 : The Infotainment System

NHTSA would require rear cameras in all vehicles by 2018.

Already becoming more or less mandatory safety feature in 2016.

Obviously needed to work almost instantly after ignition on. OEM's target was 15 seconds.



The JIRA Ticketing System

Case 2 : The Infotainment System

All 50 suppliers on one JIRA ticketing system.

Result was multi-way ping pong as suppliers rapidly redirected tickets to each other to avoid having to do any difficult debugging.

Auto-assigned JIRA ticket numbers had passed the 20,000 mark and were continuing to grow rapidly.

The OEM company culture was fascinated with “Bug Zero” as the definition of success… even though this would never be expected to occur ever by a sophisticated software development organization.

One Monday morning, open JIRA tickets had indeed dropped to “0”!

There was a brief celebration.

Then testing resumed. By Friday there were several thousand open JIRA tickets again.

Holding Supplier Teams Hostage

Case 2 : The Infotainment System

Everything else in the vehicle was ready to go. Only the infotainment system was holding up production.

We were told that 1,600 supplier factories were waiting.
All of the Infotainment suppliers were ordered to station development teams at the OEM's rural, remote development site for the foreseeable future.

Systems integration strategy : Lock everyone in a room and don't let anyone out until the system is working.

The Manager Who Loved Chaos

Case 2 : The Infotainment System

The director on the software development side was a text book example of a polar narcissist.

He LOVED attention, but he also had no capacity for empathy. All other humans were merely tools to serve his purposes.

He would come out of a meeting with the OEM – which had consisted of two hours of them screaming at him – skipping, whistling, snapping his fingers, and generally in a terrific mood. The fact that the attention was *negative* attention did not bother him in the least.

Naturally, I started working to put together a detailed and realistic recovery plan…

This manager was furious with me: ***"I need MANEUVERING ROOM!!"*** He really did not like all that awful clarity!

Back to Austin

Case 2 : The Infotainment System

The OEM was constantly pressuring the software development company to build up a credible software development center near their remote, rural design center.

The narcissist manager lived in Canada. He enjoyed his life there and did not want to relocate. On the other hand, he really did not want a strong, competent local manager/team that might jeopardize his control.

Solution: Keep promising to build such a team. Keep hiring and then ruthlessly undermining local manager. I was the 5th in succession….

Fortunately, a wonderful job appeared back in Austin just as he was preparing to eliminate me.

Take Aways

Case 2 : The Infotainment System

1 – Involuntary Systems Integration. The software development services company was forced into an untenable position with no contractual or financial leverage over the other suppliers.

2 – All-You-Can-Eat Risk Buffet. The auto maker took on far too much project risk. Many of the unstable, ambitious, and risky elements that they built into the project had little or no business value.

3 – Stakeholder Motivations. Wildly chaotic project environments attract people who love wildly chaotic project environments. These people often have a vested interest in keeping the situation chaotic.

Success with the Marching Band

Understand Organization Optimization Goals

Success with the Marching Band

Level	Name	Short Description
1	Initial	Ad hoc. Little or no process. Unpredictable results.
2	Repeatable	The team produces predictable, repeatable results - but can't explain how they do it.
3	Managed	The team has a written process - and is actually following that process.
4	Measured	The team has identified process quality metrics and is collecting data for all projects.
5	Optimizing	The project quality metrics are improving from project-to-project.

Capability Maturity Model

Developed at Carnegie Mellon University in the 1980s with Department of Defense funding. There were a number of iterations. This is sort of an "average" of the many versions.

If you are working with a "Marching Band" organization, there is a high probability that they are already at level 4 or 5 on the capability maturity scale.

As a systems integrator, this is as close to Nirvana as it gets:

1. Solid, stable processes
2. No need to battle utter chaos
3. Supplier relationships and processes already in place.

Your job is simply to understand what the organization wants optimized next and make it so.

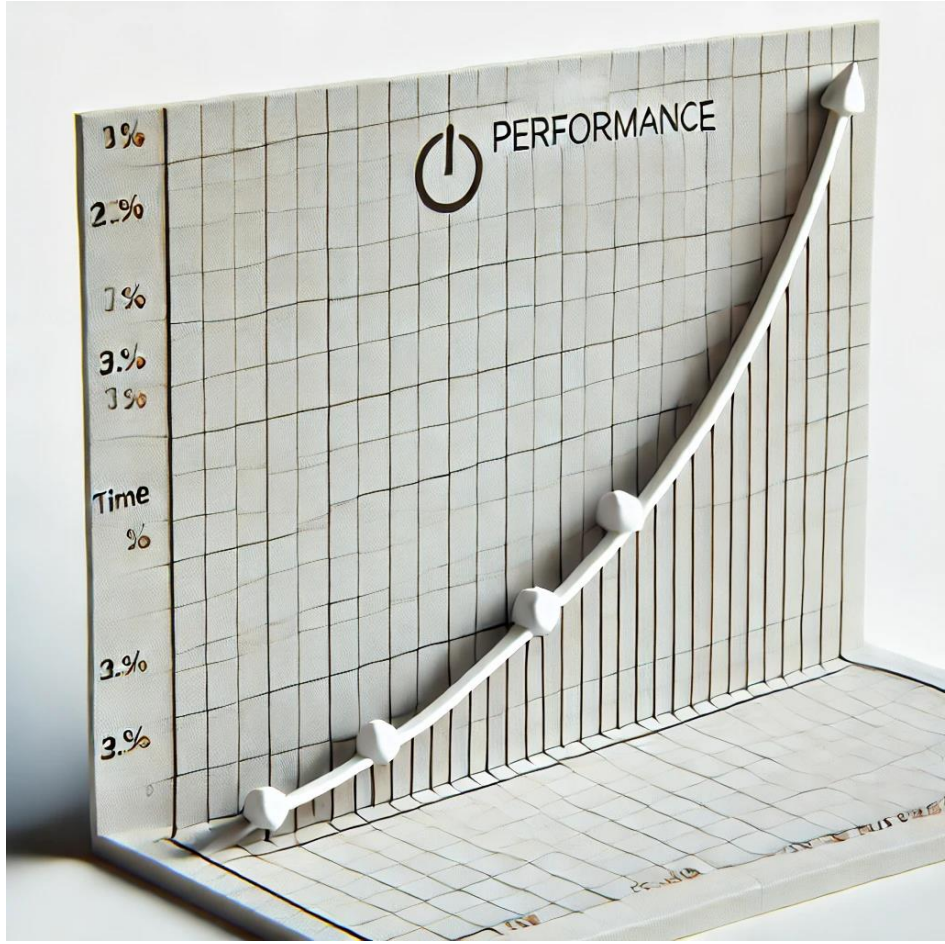
Collect/Study Historical Data

Success with the Marching Band

By definition, a CMM 4 or CMM 5 organization will have historical performance data.

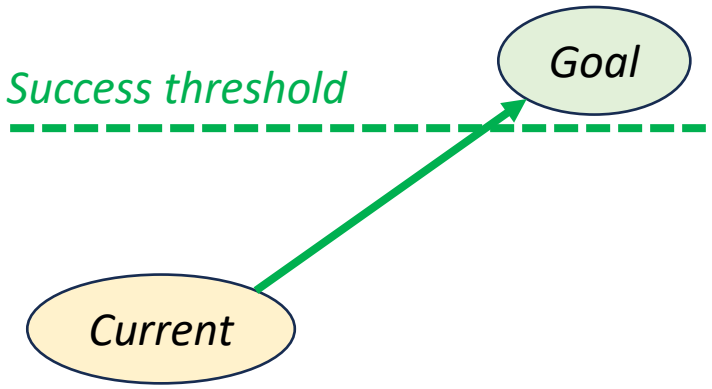
Dig in and understand the data and its implications.

Where it looks like there could be performance weak spots, dig some more. You may be able to find/extract more data that points to an area of potential performance improvement.



Identify Process Improvement Goals and Measurements

Success with the Marching Band



Pick a small number of performance indicators that you think you can improve. 1-3 would be good. Don't try to tackle 10 different things at one time.

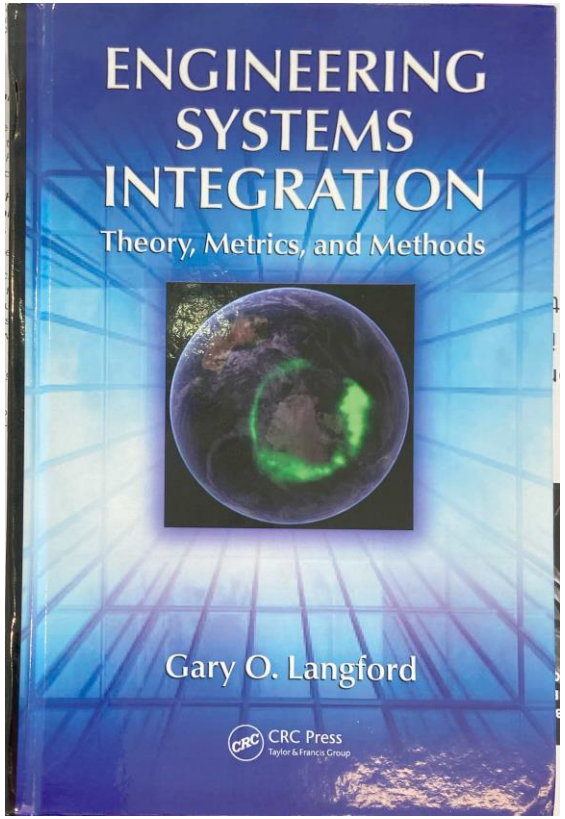
Figure out how to measure success.

Define a success threshold for each metric.

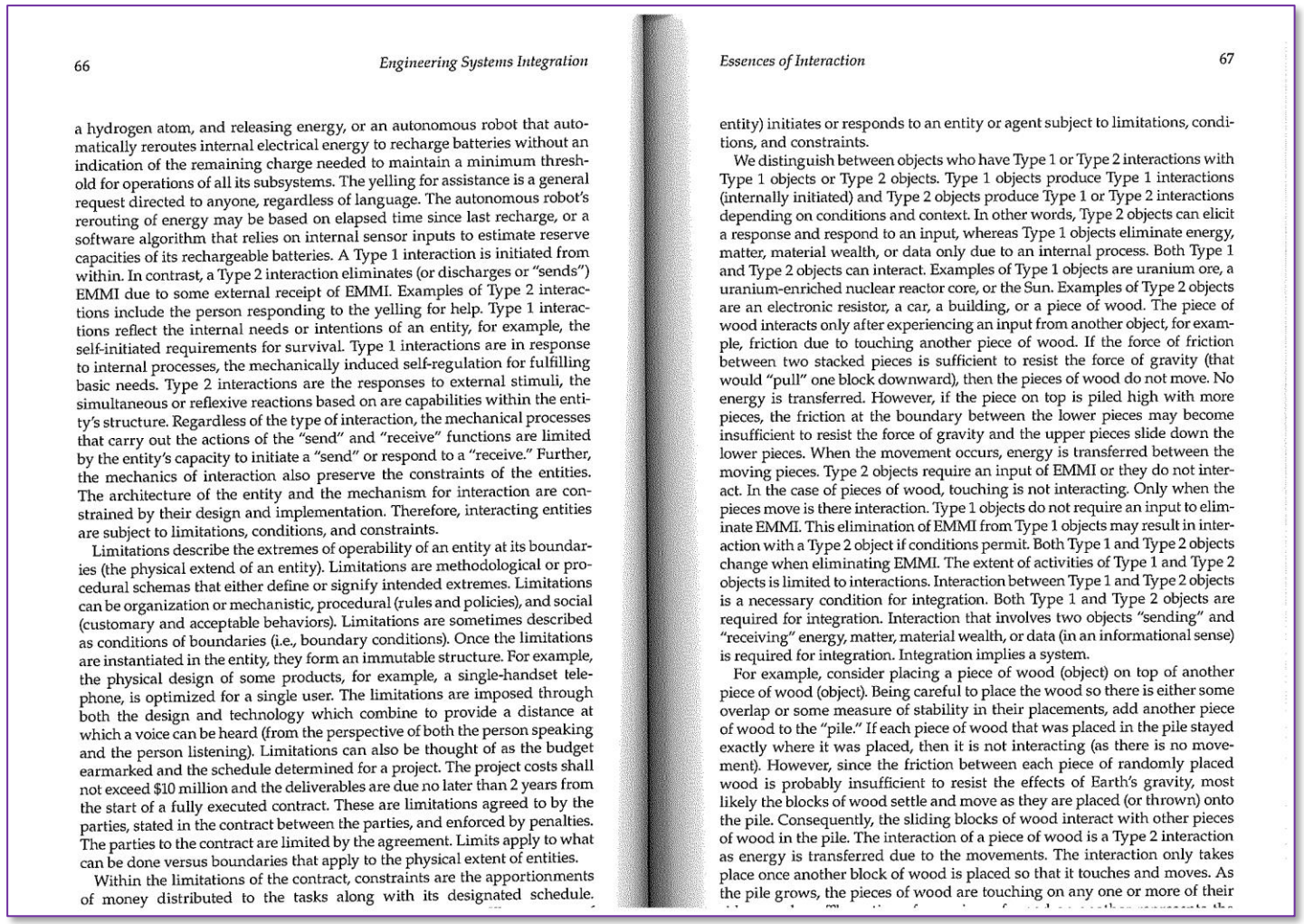
Go make it happen!

Textbooks?

Success with the Marching Band



The only book I could find on Amazon
(Purchased in 2013, checked again in 2024)



66 *Engineering Systems Integration*

a hydrogen atom, and releasing energy, or an autonomous robot that automatically reroutes internal electrical energy to recharge batteries without an indication of the remaining charge needed to maintain a minimum threshold for operations of all its subsystems. The yelling for assistance is a general request directed to anyone, regardless of language. The autonomous robot's rerouting of energy may be based on elapsed time since last recharge, or a software algorithm that relies on internal sensor inputs to estimate reserve capacities of its rechargeable batteries. A Type 1 interaction is initiated from within. In contrast, a Type 2 interaction eliminates (or discharges or "sends") EMMI due to some external receipt of EMMI. Examples of Type 2 interactions include the person responding to the yelling for help. Type 1 interactions reflect the internal needs or intentions of an entity, for example, the self-initiated requirements for survival. Type 1 interactions are in response to internal processes, the mechanically induced self-regulation for fulfilling basic needs. Type 2 interactions are the responses to external stimuli, the simultaneous or reflexive reactions based on are capabilities within the entity's structure. Regardless of the type of interaction, the mechanical processes that carry out the actions of the "send" and "receive" functions are limited by the entity's capacity to initiate a "send" or respond to a "receive." Further, the mechanics of interaction also preserve the constraints of the entities. The architecture of the entity and the mechanism for interaction are constrained by their design and implementation. Therefore, interacting entities are subject to limitations, conditions, and constraints.

Limitations describe the extremes of operability of an entity at its boundaries (the physical extend of an entity). Limitations are methodological or procedural schemas that either define or signify intended extremes. Limitations can be organization or mechanistic, procedural (rules and policies), and social (customary and acceptable behaviors). Limitations are sometimes described as conditions of boundaries (i.e., boundary conditions). Once the limitations are instantiated in the entity, they form an immutable structure. For example, the physical design of some products, for example, a single-handset telephone, is optimized for a single user. The limitations are imposed through both the design and technology which combine to provide a distance at which a voice can be heard (from the perspective of both the person speaking and the person listening). Limitations can also be thought of as the budget earmarked and the schedule determined for a project. The project costs shall not exceed $10 million and the deliverables are due no later than 2 years from the start of a fully executed contract. These are limitations agreed to by the parties, stated in the contract between the parties, and enforced by penalties. The parties to the contract are limited by the agreement. Limits apply to what can be done versus boundaries that apply to the physical extent of entities.

Within the limitations of the contract, constraints are the apportionments of money distributed to the tasks along with its designated schedule.

Essences of Interaction 67

entity) initiates or responds to an entity or agent subject to limitations, conditions, and constraints.

We distinguish between objects who have Type 1 or Type 2 interactions with Type 1 objects or Type 2 objects. Type 1 objects produce Type 1 interactions (internally initiated) and Type 2 objects produce Type 1 or Type 2 interactions depending on conditions and context. In other words, Type 2 objects can elicit a response and respond to an input, whereas Type 1 objects eliminate energy, matter, material wealth, or data only due to an internal process. Both Type 1 and Type 2 objects can interact. Examples of Type 1 objects are uranium ore, a uranium-enriched nuclear reactor core, or the Sun. Examples of Type 2 objects are an electronic resistor, a car, a building, or a piece of wood. The piece of wood interacts only after experiencing an input from another object, for example, friction due to touching another piece of wood. If the force of friction between two stacked pieces is sufficient to resist the force of gravity (that would "pull" one block downward), then the pieces of wood do not move. No energy is transferred. However, if the piece on top is piled high with more pieces, the friction at the boundary between the lower pieces may become insufficient to resist the force of gravity and the upper pieces slide down the lower pieces. When the movement occurs, energy is transferred between the moving pieces. Type 2 objects require an input of EMMI or they do not interact. In the case of pieces of wood, touching is not interacting. Only when the pieces move is there interaction. Type 1 objects do not require an input to eliminate EMMI. This elimination of EMMI from Type 1 objects may result in interaction with a Type 2 object if conditions permit. Both Type 1 and Type 2 objects change when eliminating EMMI. The extent of activities of Type 1 and Type 2 objects is limited to interactions. Interaction between Type 1 and Type 2 objects is a necessary condition for integration. Both Type 1 and Type 2 objects are required for integration. Interaction that involves two objects "sending" and "receiving" energy, matter, material wealth, or data (in an informational sense) is required for integration. Integration implies a system.

For example, consider placing a piece of wood (object) on top of another piece of wood (object). Being careful to place the wood so there is either some overlap or some measure of stability in their placements, add another piece of wood to the "pile." If each piece of wood that was placed in the pile stayed exactly where it was placed, then it is not interacting (as there is no movement). However, since the friction between each piece of randomly placed wood is probably insufficient to resist the effects of Earth's gravity, most likely the blocks of wood settle and move as they are placed (or thrown) onto the pile. Consequently, the sliding blocks of wood interact with other pieces of wood in the pile. The interaction of a piece of wood is a Type 2 interaction as energy is transferred due to the movements. The interaction only takes place once another block of wood is placed so that it touches and moves. As the pile grows, the pieces of wood are touching on any one or more of their

Sample of the content from this book.

I found this book difficult to read.

INCOSE Wiley Online Library

Success with the Marching Band

Systems integration and architecting: An overview of principles, practices, and perspectives
Andrew P. Sage, Charles L. Lynch
First published: 11 January 1999 https://doi.org/10.1002/(SICI)1520-6858(1998)1:3<176::AID-SYS3>3.0.CO;2-L

Systems Integration: Key Perspectives, Experiences, and Challenges
Azad M. Madni, Michael Sievers
First published: 29 May 2013
https://doi.org/10.1002/sys.21249

System of Systems Integration: Key Considerations and Challenges
Azad M. Madni, Michael Sievers
First published: 01 July 2013 https://doi.org/10.1002/sys.21272

Integration principles for complex systems
Joshua Logan Grumbach, Lawrence Dale Thomas
First published: 28 September 2020 https://doi.org/10.1002/sys.21554

Systems integration implications of component reuse
Joshua Logan Grumbach, Lawrence Dale Thomas
First published: 09 August 2022 https://doi.org/10.1002/sys.21636

Quantitative validation of complex systems integration principles
Joshua Logan Grumbach, Lawrence Dale Thomas
First published: 19 September 2022 https://doi.org/10.1002/sys.21641

The INCOSE Wiley Online Library is much better!
https://incose.onlinelibrary.wiley.com/

If you login via the main INCOSE website downloads are free for INCOSE members.

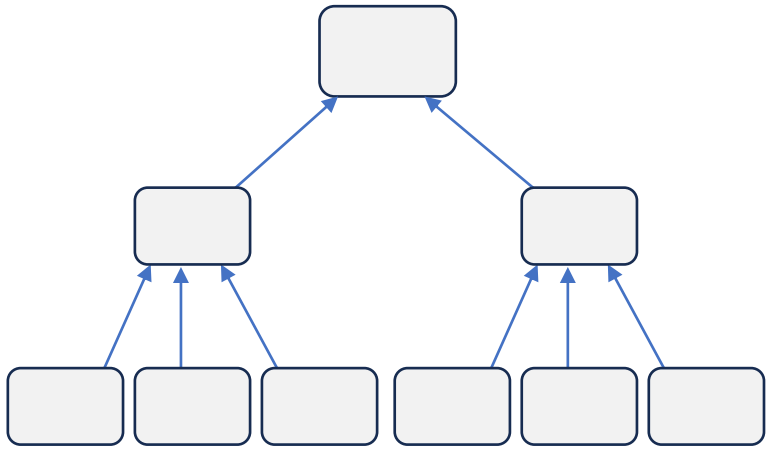
I only found six articles, but they are all very good.

- Easy to read
- Outstanding interesting case studies
- Solid conceptual principles and frameworks

This material should be more than enough to get you on your way to helping your “Marching Band” further optimize its processes!

Example – Hierarchical Integration

Success with the Marching Band



Basic idea is to first integrate small clusters, then integrate clusters into larger clusters, and so on.

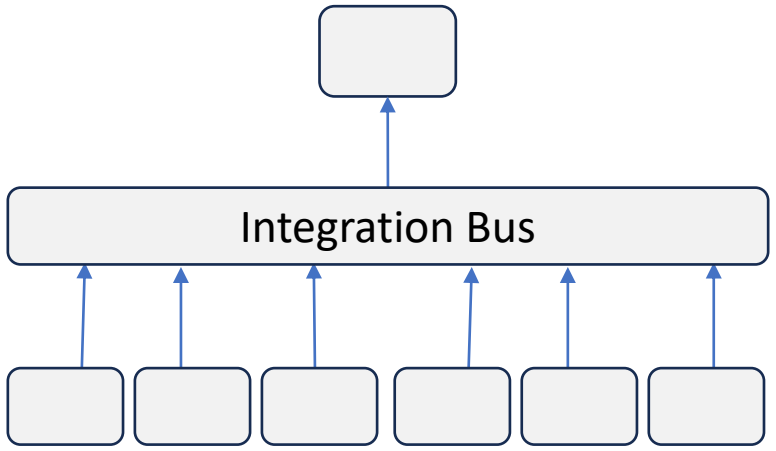
The Wiley papers contain some semi-mathematical reasoning about why this approach will always be faster than random/parallel integration.

Generally this is a solid approach with the side benefit that it forces analysis of your architectural structure to identify the clusters.

However, this approach is very susceptible to the “Late Dinner Guest” problem described on slide 25. One or two late or problematic subsystems can wreck the entire plan.

Example – Integration Bus

Success with the Marching Band



Basic idea is that one subsystem or technology serves as the “integration bus” that every thing else attaches to.

The 1999 paper mentions CORBA as a candidate. CORBA is still around, but it never quite took the world by storm.

Done well, this approach can alleviate the pressure to integrate in a rigid order. If feasible, this approach can help manage the “Late Dinner Guest” problem.

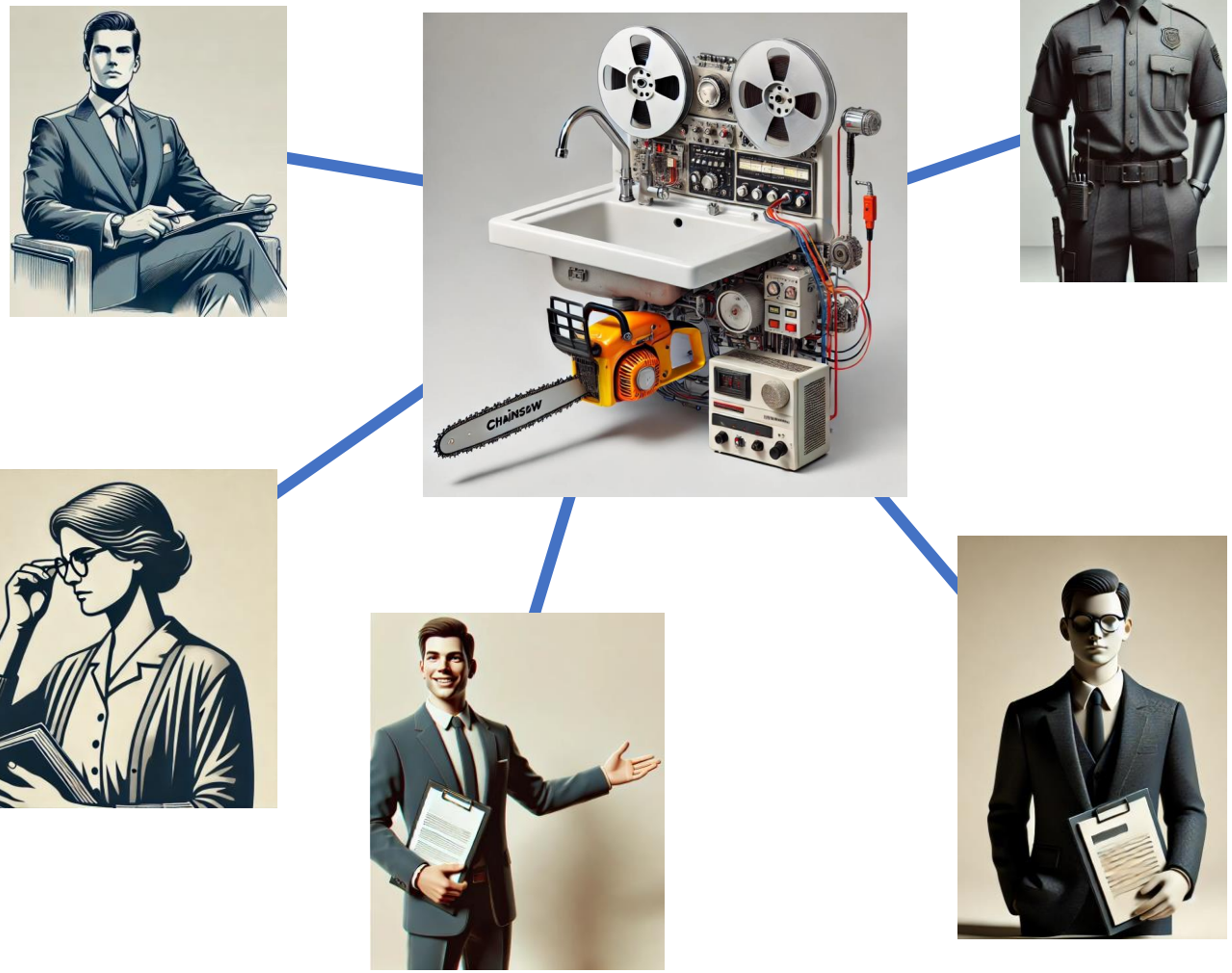
One downside is that the integration bus technologies (like CORBA) can be quite complex and difficult to implement in their own right.

(This idea is popular for digital engineering tool chain integration too with several tool suppliers providing products of this type.)

Success with the Goat Rodeo

Understand Stakeholder Motivations

Success with the Goat Rodeo



The Goat Rodeo organization will usually be at a CMM 0.2 or lower level of maturity.

No optimization here. Simple survival.

The most important first step is to dig in and get to understand the stakeholders.

- Who are they?
- Are there “hidden” stakeholders?
- Who controls the money?
- Who makes the decisions?
- Who influences?
- Who excels at throwing their mother under the bus?
- What is each stakeholder afraid of?
- What goals (including selfish goals) does each stakeholder have?

Collect/Study Historical Data

Success with the Goat Rodeo

Discretely find out what kind of data you can obtain for the organization's historical performance.

Look for mistakes, setbacks, and cost overruns.

Triage the Problem

Success with the Goat Rodeo

Triage the problems.

How much time and resources do you have?

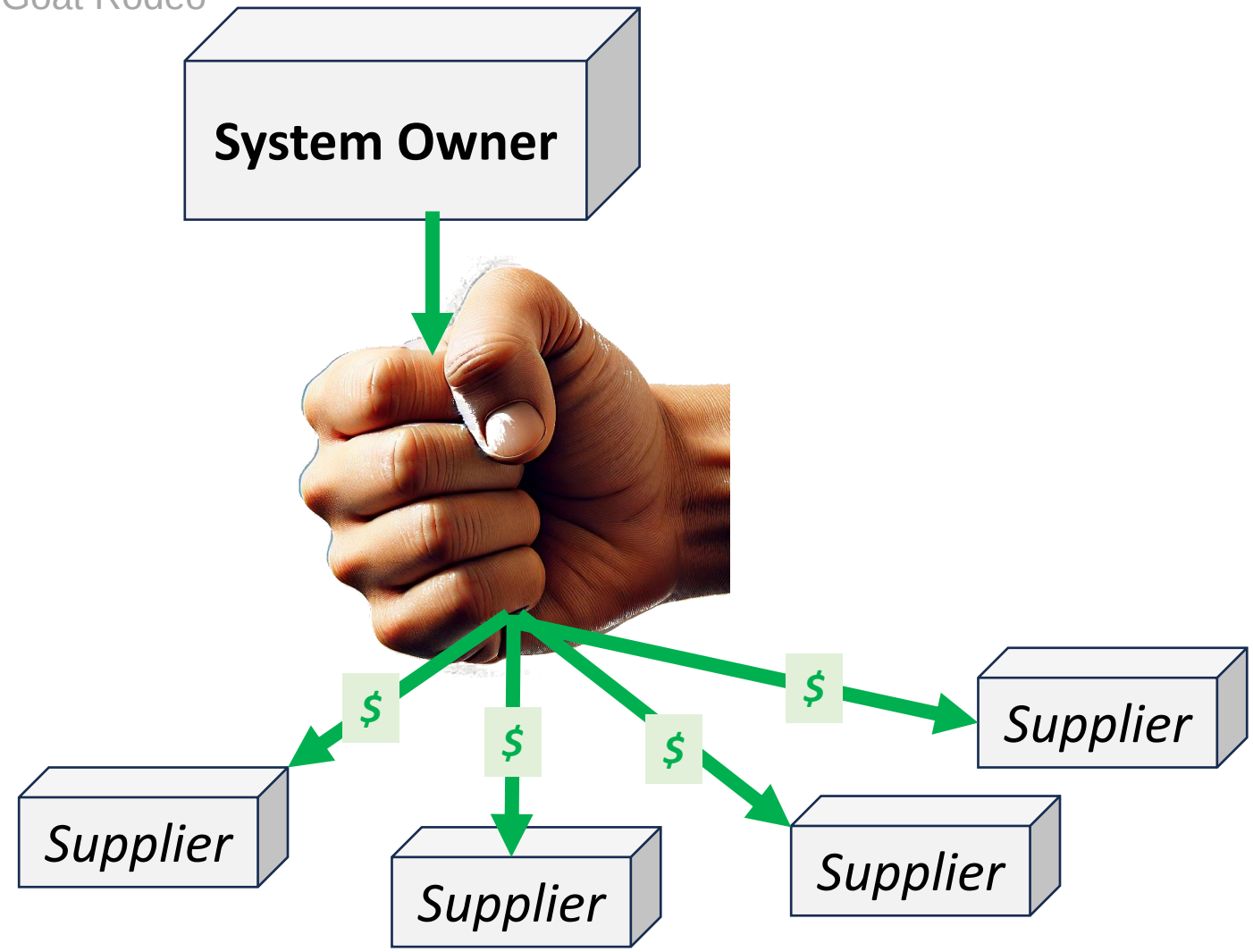
Which problems will be tolerable if left to fizzle along on their own?

Which problems will be so difficult to solve that they will consume you and you will fail anyway?

Where will your efforts yield real benefits?

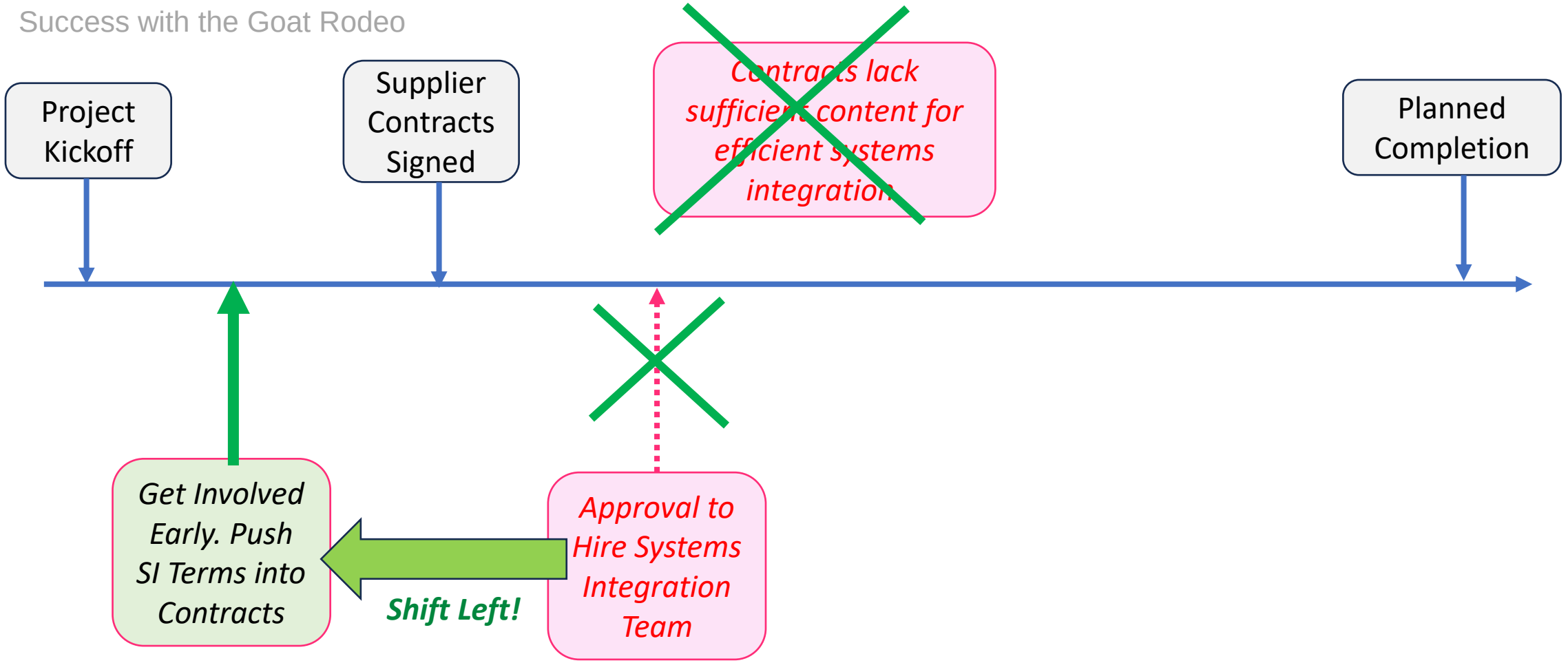
Fight to Get Control of Payment Stream

Success with the Goat Rodeo



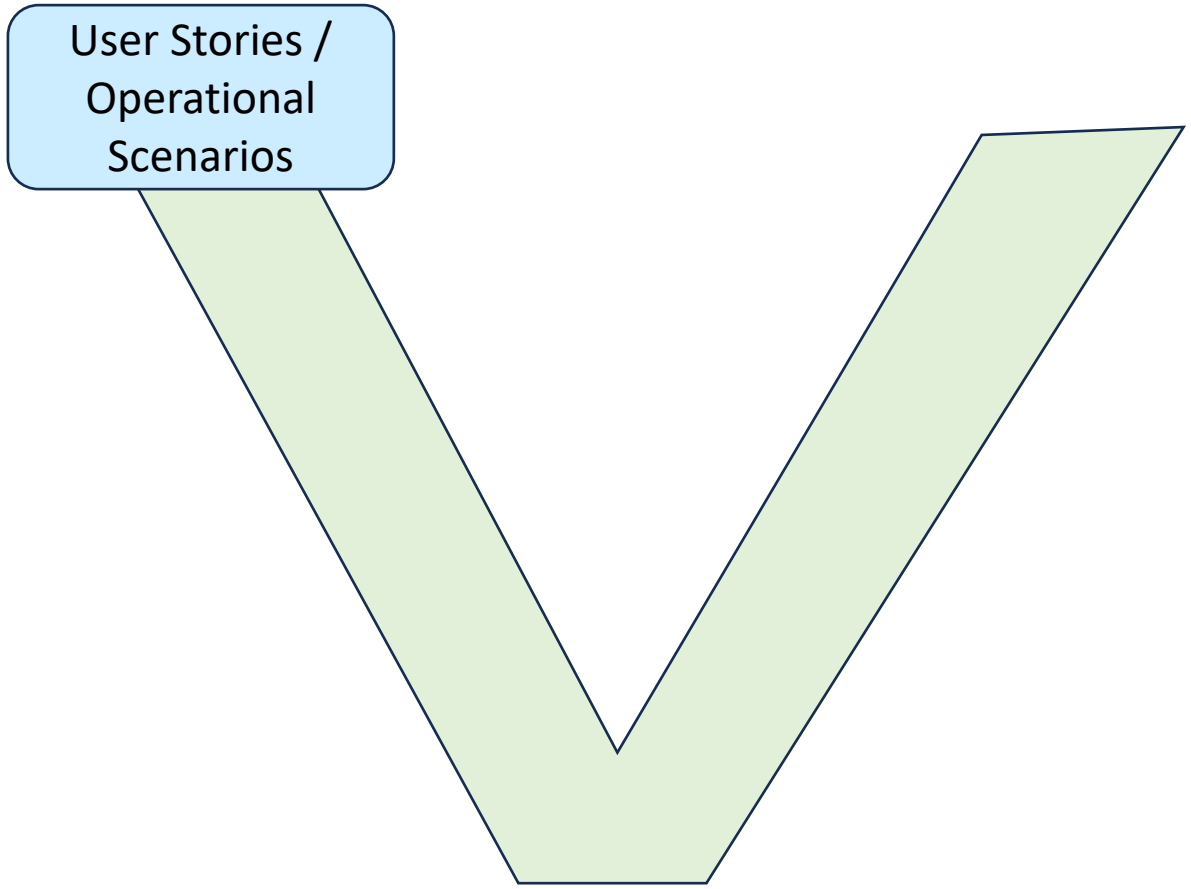
Fight to Get Content into the Contracts

Success with the Goat Rodeo



Fight to Get Approval for a Set of User Stories

Success with the Goat Rodeo



These are often a *"The Dog Ate Our Homework"* item…

Fight hard to get a set of user stories written down and approved by the stakeholders.

Clear User Stories will be crucial for imposing order on the rest of the systems integration process.

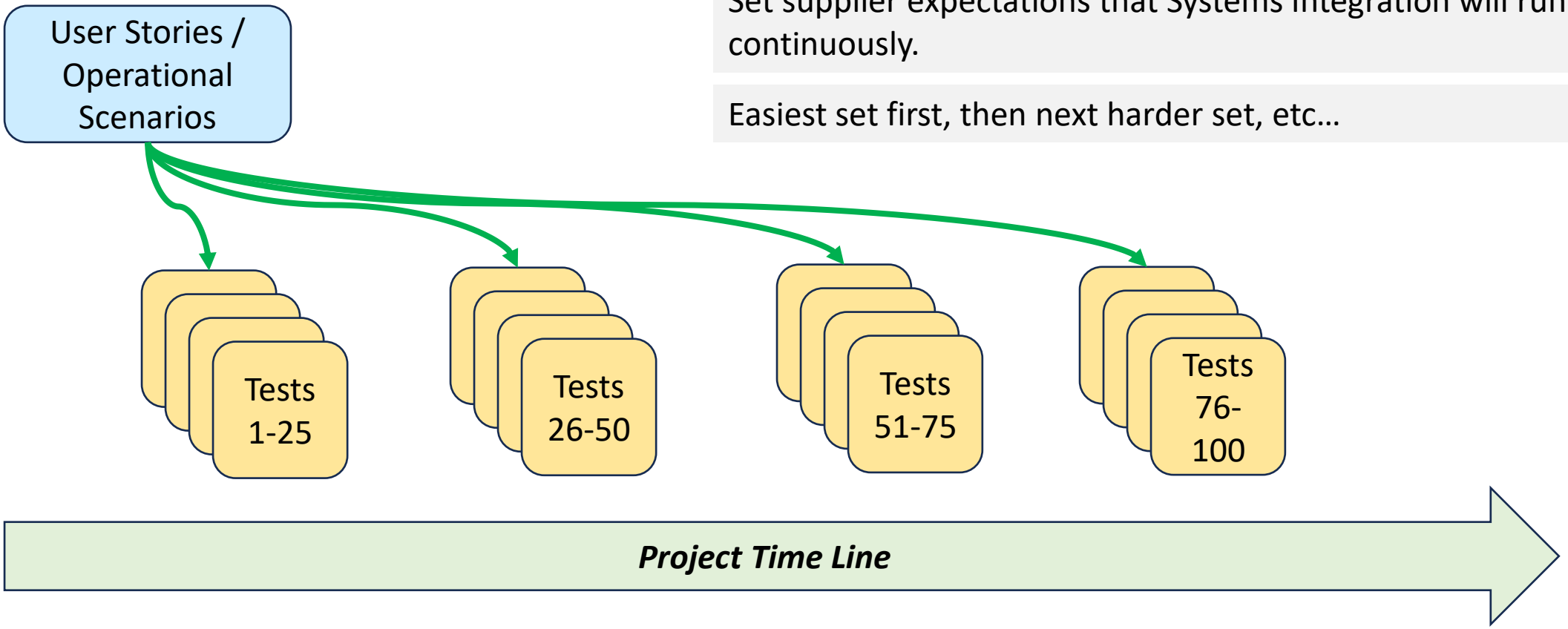
Create and Bucket Test Cases for Clear Progress Points

Success with the Goat Rodeo

Prevent procrastination

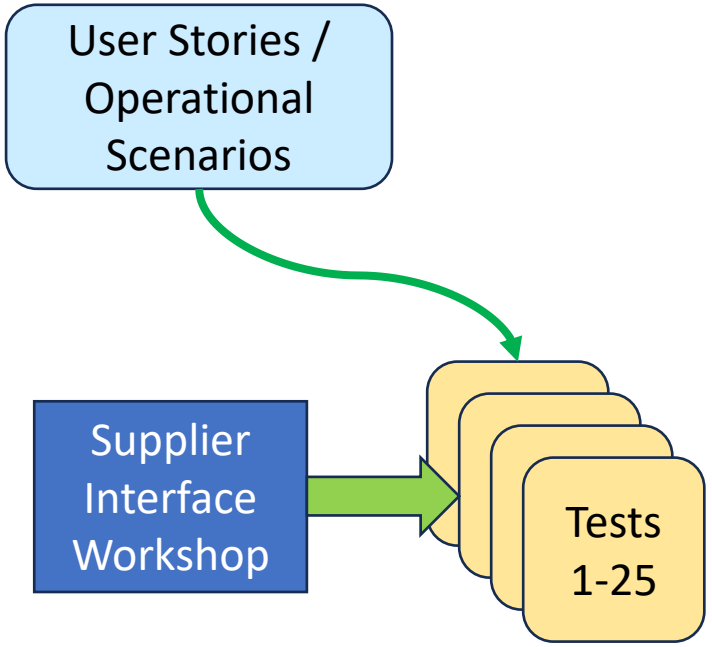
Set supplier expectations that Systems Integration will run continuously.

Easiest set first, then next harder set, etc…



Hold Interface Workshops for Each Round of Test Cases

Success with the Goat Rodeo



Schedule supplier interface workshops for each bucket of tests.

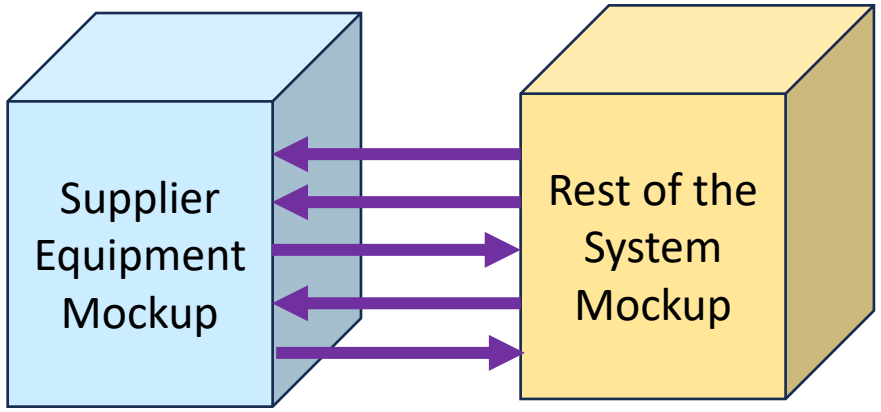
Force suppliers to sit in a room and thrash out the expected sequences of communication for each user story.

Consider creating and publishing a MBSE model to document all of the expected sequences.

Project Time Line

Invest in Software Mockups

Success with the Goat Rodeo



After the suppliers work out the sequences, throw together two software mockups for each supplier.

The two mockups exchange messages as agreed by the suppliers.

Provide these to the suppliers for test and debug.

Fastest, dirtiest, quickest thing you can throw together.

- Avoid elaborate architecture exercises.
- NOT full simulations. Just send and receive messages.

Require suppliers to demonstrate all test cases in each bucket by the scheduled checkpoint.

If possible, visit the supplier's sites and make them demonstrate live and in person.

Communicate, Communicate, Communicate!

Success with the Goat Rodeo

Don’t allow anyone to go radio silent.

Weekly status calls with every supplier.

Demand demonstrated progress against the test cases.

Visit supplier sites in person and inspect.

Keep communicating!

Questions?

Thank You!

System Strategy, Inc.
1221 Bowers St, #50
Birmingham, MI 48012

T: 844.SYSTEMX
F: 844.SYSTEM0

info@systemxi.com

David Hetherington
Austin, Texas

+1 (512) 695-1365
dhetherington@systemxi.com