USC University of Southern California

USC CSSE

Expediting Systems Engineering in a System of Systems

Presented at 16th ANNUAL PSM USERS' GROUP MEETING

Jo Ann Lane and Supannika Koolmanojwong
Center for Systems and Software Engineering
University of Southern California

Overview

- Characterize "expediting"
- Overview of current research
- Approaches for "expediting"
  - Single system
  - Systems that participate in one or more systems of systems (SoS)
  - SoS capabilities
- Related technical debt issues
- Understanding "expediting" and "technical debt trades using cost models

What Does it Mean to "Expedite Systems Engineering"?

- Expedite "systems engineering" or "system development"?
  - Most are interested in "system development": *Capability development schedule from concept to delivery*
  - Some will include enhancement, maintenance, retirement
- For our research (and this presentation), includes
  - Systems engineering
  - Development and procurement activities
    - Hardware
    - Software
  - Evolution/enhancement
  - Maintenance
  - Retirement

Early decisions can affect ability to expedite later….

General Ways to "Expedite"

- Minimal engineering/quick solutions
- Minimal features
- Commercial-off-the-Shelf (COTS) solutions
- Lean approach
  - Eliminate non-value adding activities
  - Reduce wait times
- Pacing
  - Go slow to establish good
    - Foundation
    - Architecture
    - Interfaces
    - Relatively low complexity
  - Then go fast

Difficult in SoS environment since "foundations" seldom formally developed…

Balancing "Expedited Engineering" in the SoS Environment

- Trades to consider when "expediting"
  - Long term affordability
  - Flexibility/adaptability for meeting future needs
  - Desired level of performance/speed/throughput
  - Maintainability
  - Securability
  - … and others
- Trades may
  - Reduce future flexibility
  - Result in
    - Degradation of existing capabilities
    - System limitations
    - Later rework

With competing trades at the single system at SoS levels

Depending on the situation/need, it may be OK to incur technical debt….

Tradespace Example: System Flexibility

- Goal of “flexibility” is to go beyond quick solution to build in flexibility that will allow system to
  - Easily evolve in the future to meet future (often unknown) needs
  - Interoperate with future systems (e.g., in one or more SoS environments)
- Must balance “flexibility” with “complexity”
  - Performance issues may result if system tries to be “everything for everyone”
- Ways to evaluate flexibility
  - Total ownership costs
  - Option analyses using Monte Carlo techniques

Expediting Development and Increasing Value through Flexibility*

- Flexibility in design
  - Routinely improves expected value by 25% or more
  - Enables system to
    - Avoid future downside risks
    - Take advantage of new opportunities
  - Often reduces initial capital expenditures
    - Greater expected value at less cost
    - Enables manager to better control the risks
    - Substantial increases on the return on investment
- "Sweet-spot" found through Monte Carlo analysis of business options
  - Identifies how much engineering/system performance/system capacity is enough
  - Allows future decisions/investments to be made when more is known about the future
- Types of flexibility to explore depend on the context

* Richard de Neufville and S. Scholtes, *Flexibility in Engineering Design*, The MIT Press, Cambridge MA, 2010.

USC University of Southern California

USC CSSE

Examples of Inadequate Flexibility

- Joint Tactical Radio System[1]
  - Too many waveforms led to poor performance, heating problems
  - Non-conformance to architecture standards reduced portability of waveforms across platforms
- Future Combat Systems[2]
  - Planned to be everything for everyone
  - Due to schedule pressures, foundations/core technologies not sufficiently matured
  - Did not anticipate the changing battle environment
    - Shift from conventional warfare to counter-terrorism

1. http://en.wikipedia.org/wiki/Joint_Tactical_Radio_System
2. http://en.wikipedia.org/wiki/Future_Combat_Systems

Examples of Flexible Engineering

- Global Positioning System (GPS)[1]
  - Evolved to general purpose technology (beyond military missions): commercial vehicles, cell phones, other handheld devices
  - But missed opportunity to be able to increase return on investment by charging commercial users for usage
- Unmanned Aerial Vehicles (UAVs)[1]
  - Transition from slower surveillance systems with modest payloads to faster weapons systems that can escape anti-aircraft fire
  - Transition from military surveillance to Forest Service surveillance
- Littoral combat ships[2]
  - Ability to quickly reconfigure for multiple missions (surveillance, weapons, scientific, humanitarian aid)

1. Richard de Neufville and S. Scholtes, *Flexibility in Engineering Design*, The MIT Press, Cambridge MA, 2010.
2. http://en.wikipedia.org/wiki/Littoral_combat_ship

USC University of Southern California

USC CSSE

Recent Research Overview

Capability Options

New system or system of system(s)
New procedures for using existing systems
Changes to existing system or SoS
- Some robust, well integrated
- Others very fragile, close to end of life
- Which to invest in/which to retire

Existing vs. new technologies
How much, how fast, how accurate, etc. is enough

Key Approaches for Incorporating Flexibility

Employ open architectures
Design for reuse
Develop/use product lines
Standard interfaces, protocols, services, data
Options-based design
Incremental commitment

What is useful, affordable, and sustainable?

Key Approaches for Expedited Engineering

Commercial-Off-the-Shelf (COTS) Products
Investment in product-line architectures
Reuse of existing systems/components
Repurposing existing systems/components
Value-stream focus (lean)
Going fast in general (crisis response)
Single purpose architecture

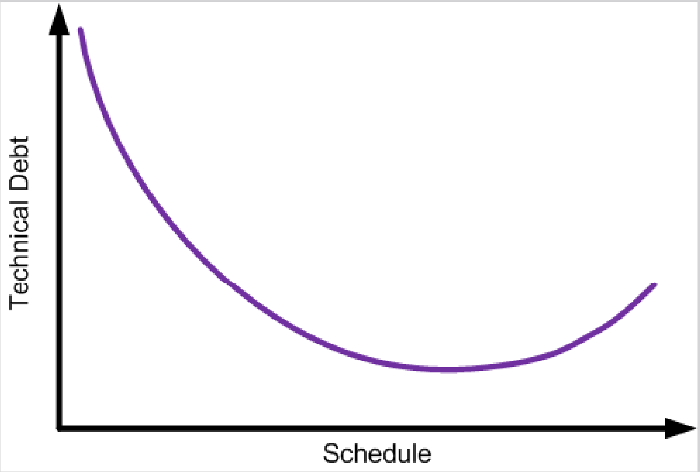
Using the right people

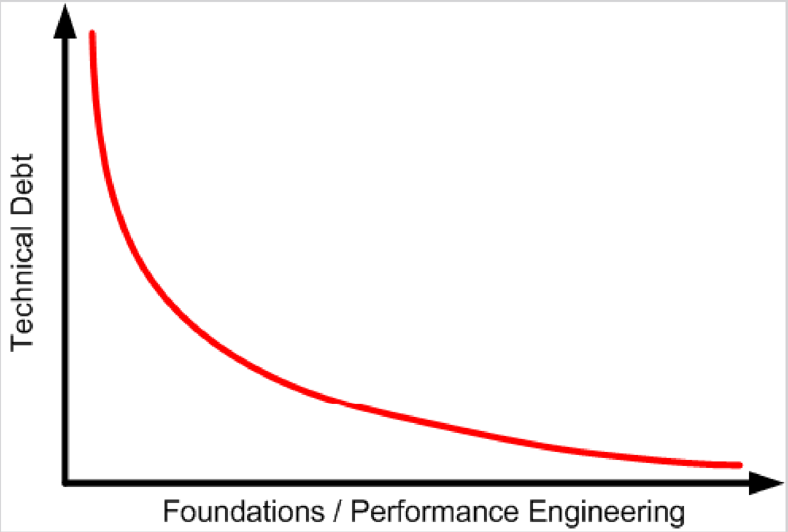
Common Causes of Technical Debt

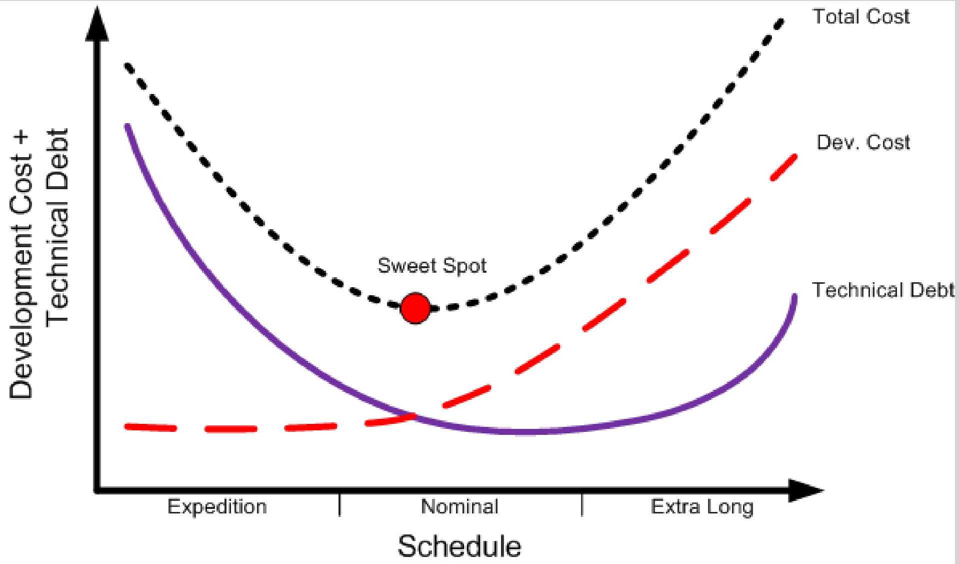
Pressures to compress schedule
Lack of requirements understanding
Lack of system understanding
Inflexible architectures/software
Overly complex design/implementation
Delayed defect resolution
Inadequate testing
Lack of current documentation
Parallel development in isolation
Delayed refactoring

Can be extended to incorporate other "-ilities"…

Single System Development Perspective



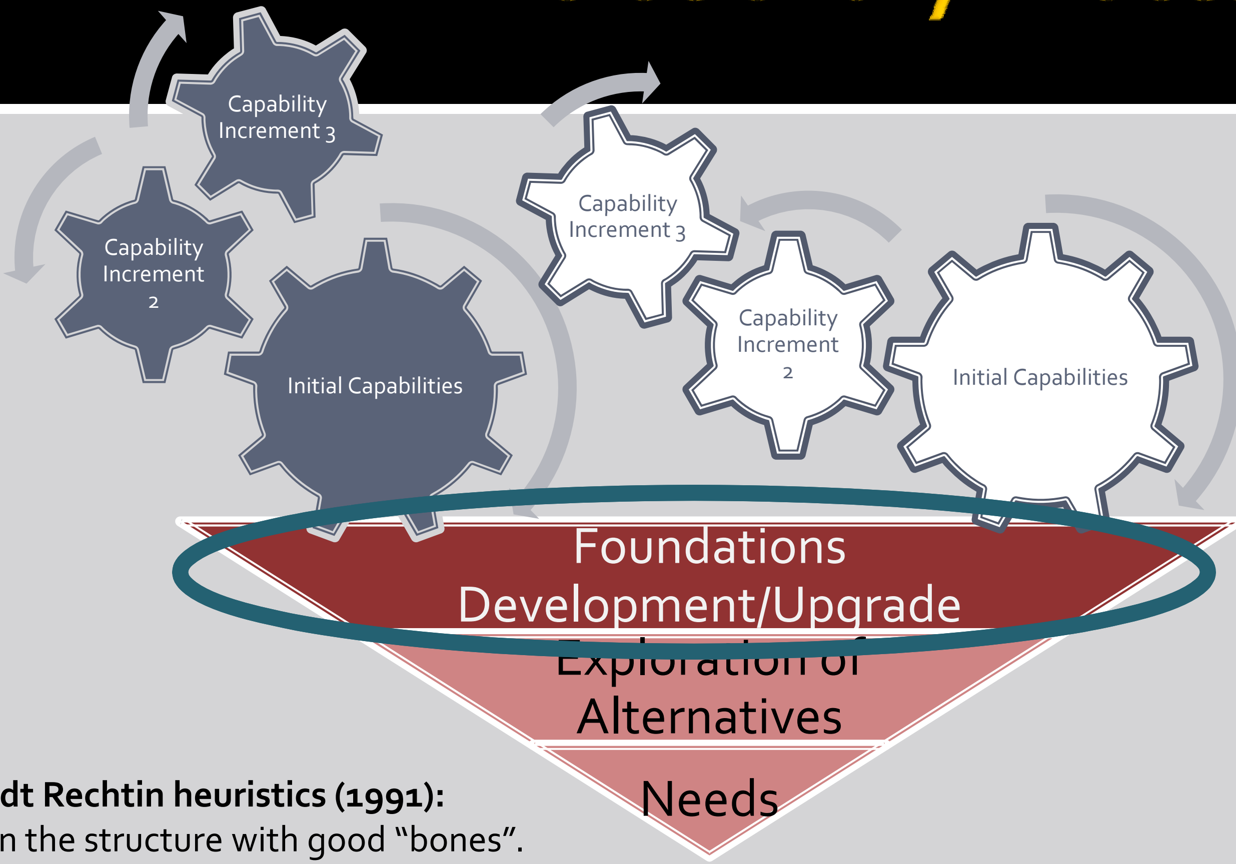




Choices driven by potential

- *Market share*
- *Future opportunities*
- *Technical debt*
- *Cost of failure to provide needed capability*

Evolutionary Processes



Some related Eberhardt Rechtin heuristics (1991):

Good Bones: Design the structure with good "bones".

KISS: Keep it simple, stupid (and other variations).

Extreme Requirements: Extreme requirements should remain under challenge throughout system design, implementation, and operation.

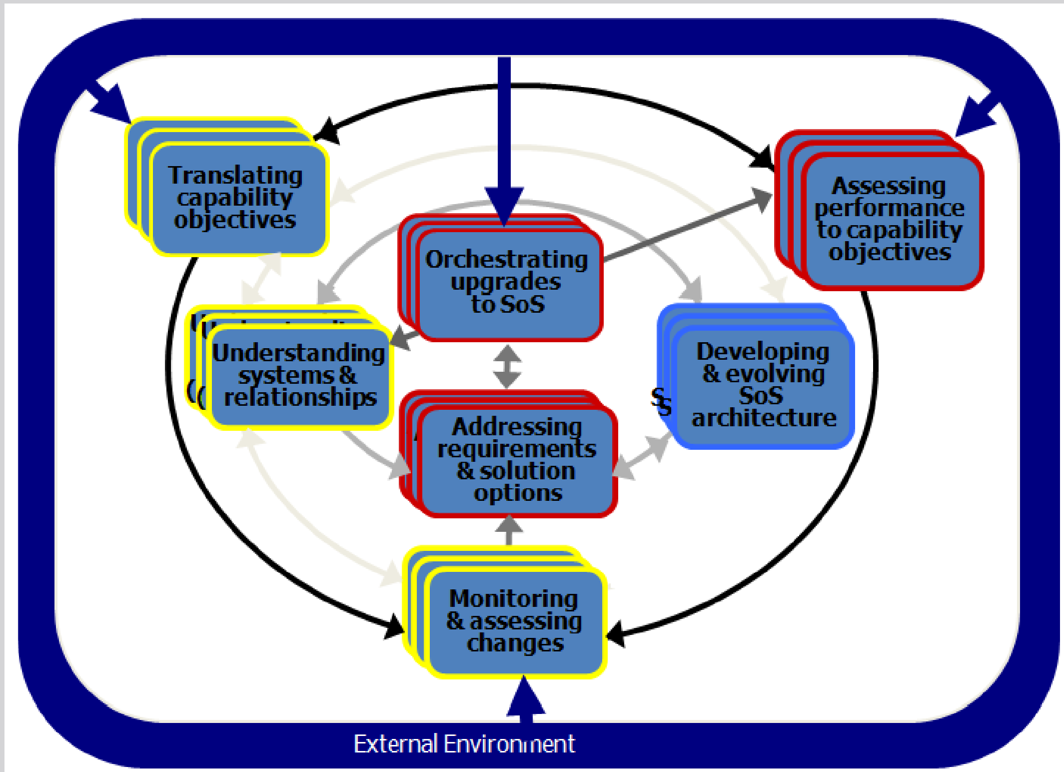
Aggregation: Choosing the appropriate aggregation of functions is critical in the design of systems.

Cost and Schedule: …by the time of the first design review, performance, cost, and schedule will have been predetermined. One might not know what they are yet, but, to first order, all the critical assumptions and choices will have been made that determine those parameters.

USC University of Southern California

USC CSSE

System of Systems Engineering Activities



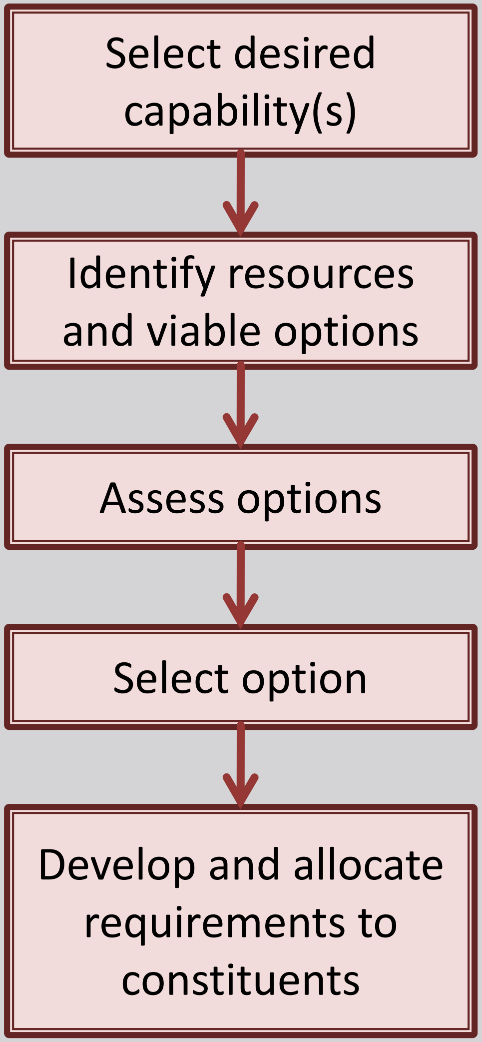
SoSE Guidebook* [1] view based on interviews and analysis of 18 DoD SoSs:

- Communications systems
- Command and control systems
- Integrated combat systems
- Ballistic missile defense systems
- Intelligence information systems
- Space-related systems

* http://www.acq.osd.mil/sse/docs/SE-Guide-for-SoS.pdf

SoS Capability to Requirements Engineering

- **Capability:** High level description of a need that is relatively independent of the constituent systems
- **Goal:** Starting with the identification of a needed capability, how to identify and assess options for decomposing capability into a set of allocated requirements that will eventually result in a testable capability



USC University of Southern California

USC CSSE

Capabilities Engineering

Identify resources:
SysML Objects

Determine options:
Responsibility/dependability/risk modeling

Assess options:
- Net-centricity/interoperability matrices
- Use cases to evaluate how
- Trades with respect to data fusion needs/formats
- Cost model estimates for most viable options

Select option

Develop and allocate requirements to constituents

Strongly suggests that data standards and common protocols facilitate expedited engineering…

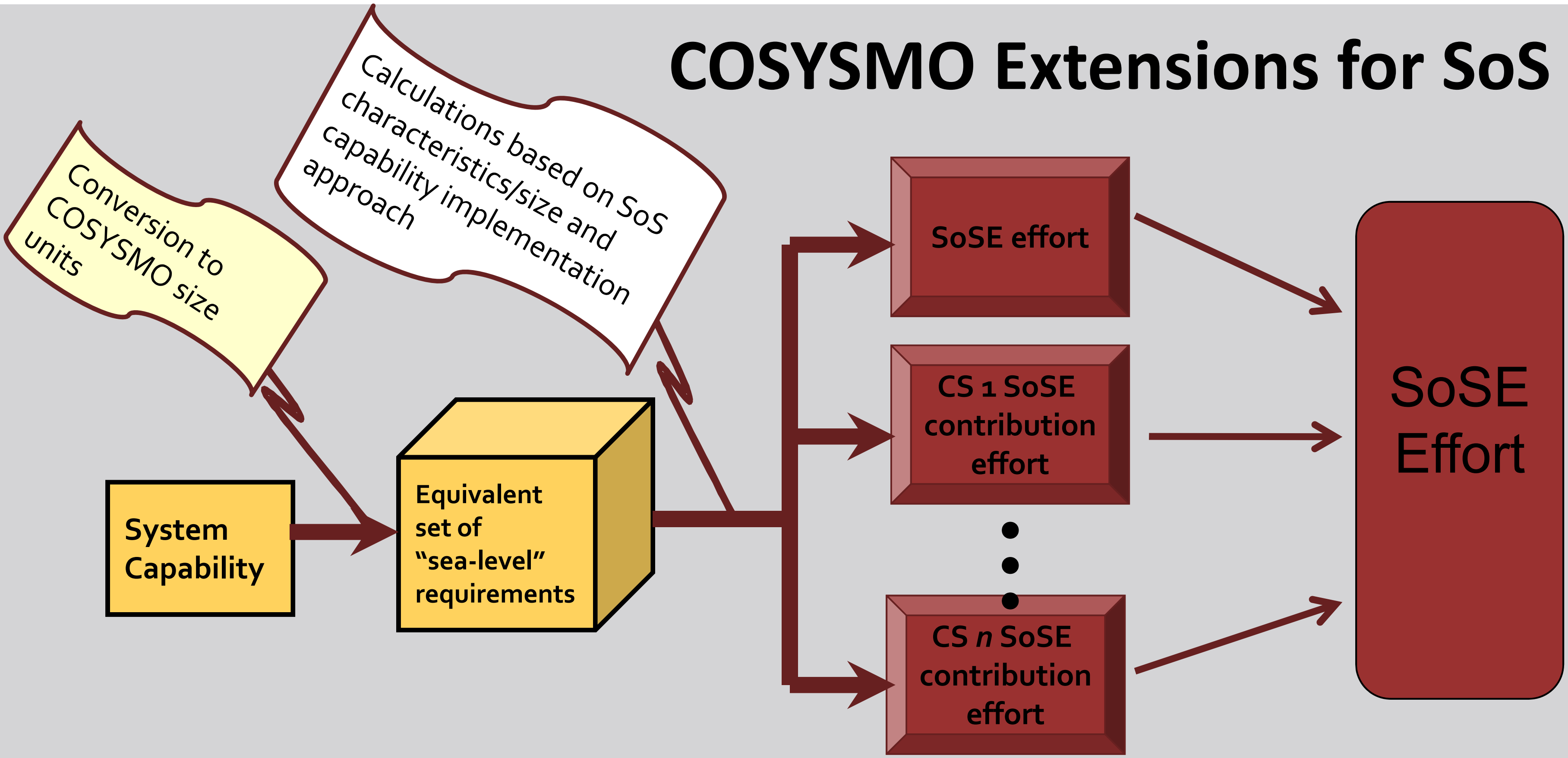
USC University of Southern California

USC CSSE

Using Cost Models to Support SoS Trades: Examples and Case Studies

Estimating SE Costs for SoS Capability Options

COSYSMO Extensions for SoS



Applies reuse factors, different cost factors for each engineering organization at each system level, and diseconomy of scale for SoS and CS-level requirements implemented in the same upgrade cycle….

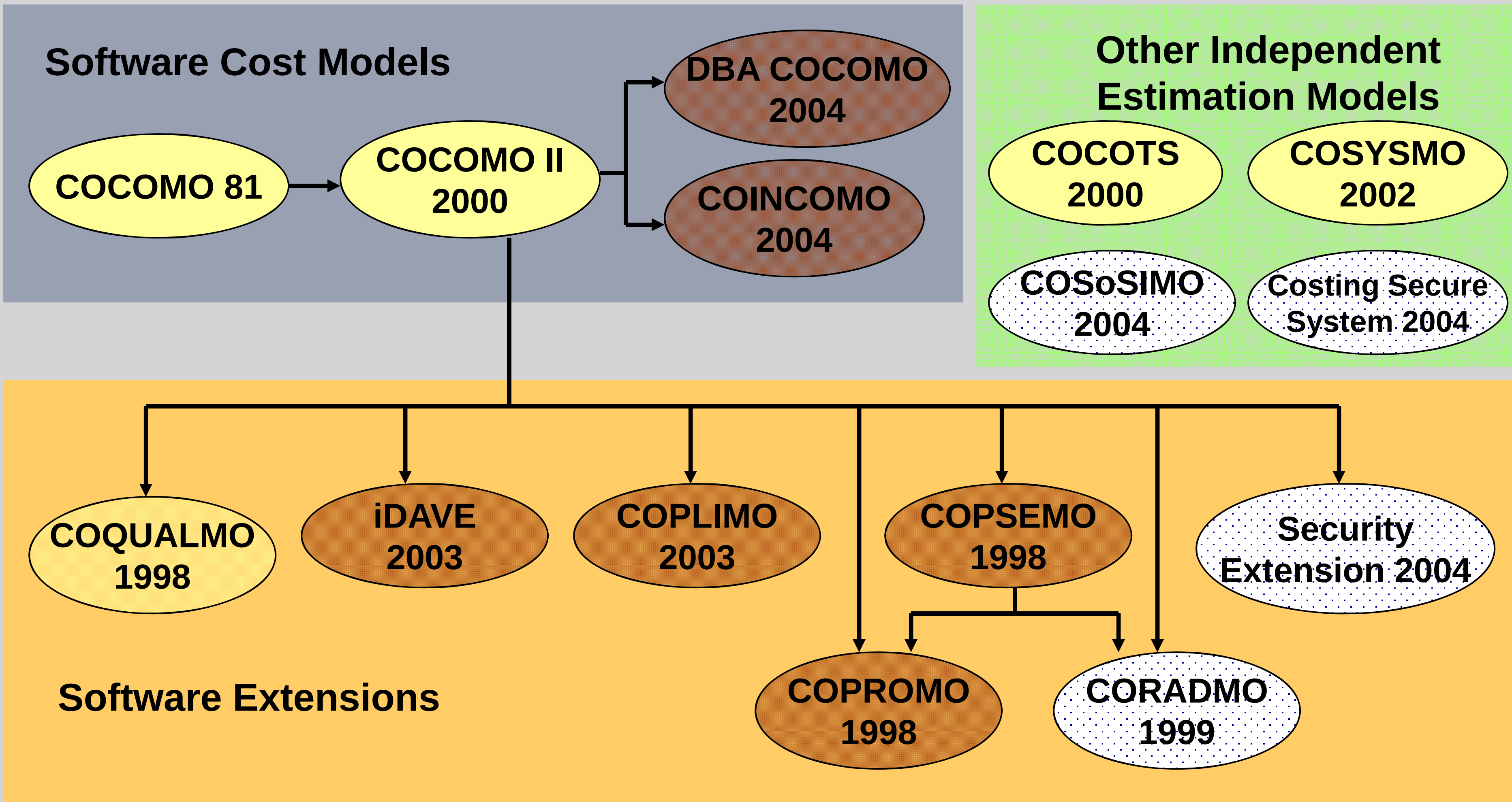
Total SoS Costs

- SE for SoS capabilities
- SE for single system capabilities
- Software development
  - Single systems
  - SoS infrastructure
- Investments in
  - Flexibility
  - Other "ilities"
- Maintenance including technology upgrades
- Savings from expedited development
- Technical debt realized from shortcuts

USC University of Southern California

USC CSSE

COCOMO Models* to Support Tradespace Analyses

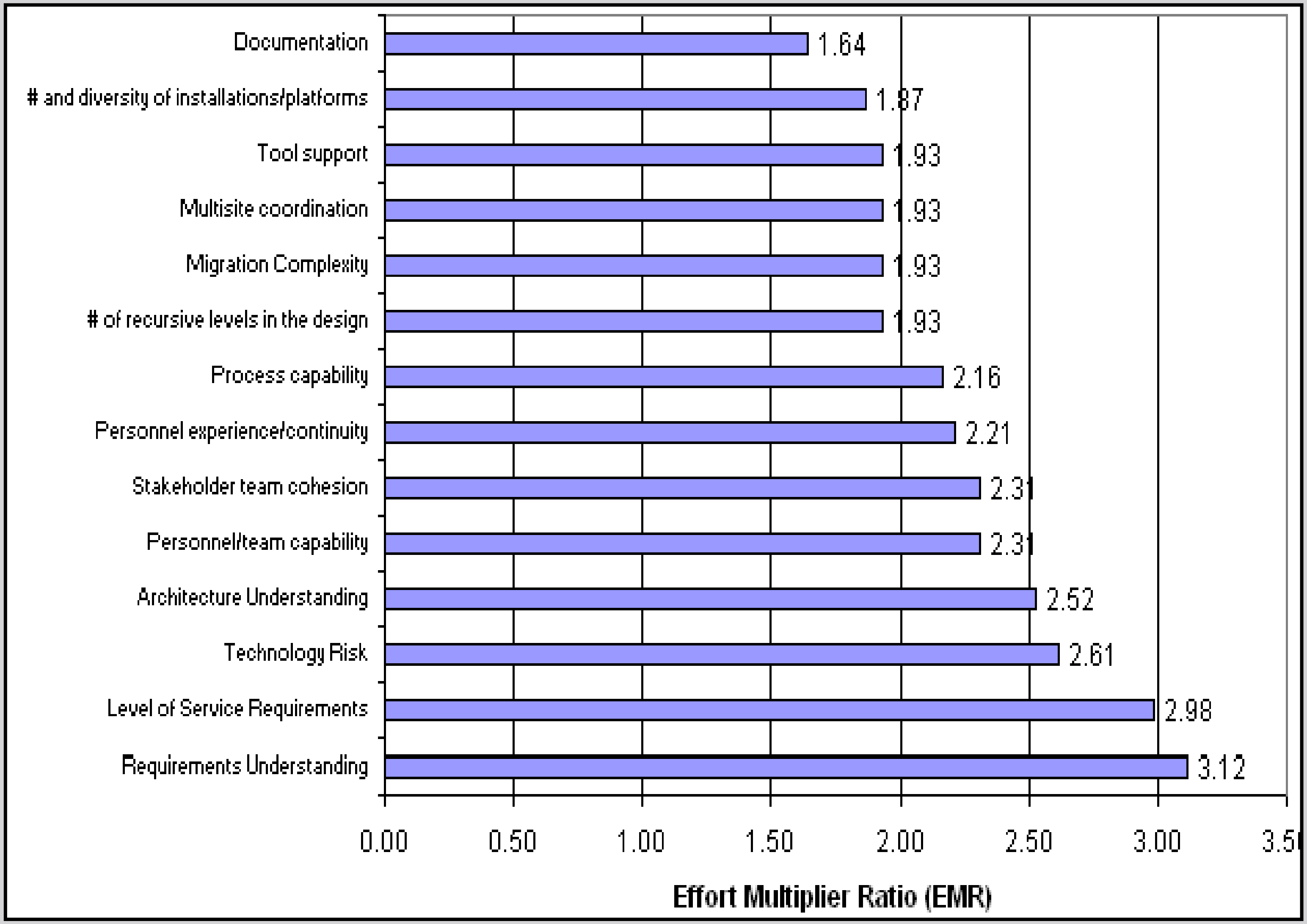


* Circa 2005

Schedule Estimation

- Schedule estimation for COSYSMO and COCOMO:
  - Cube root function of effort
- Observations
  - Reducing system/software size will reduce schedule
  - Reducing overall effort through cost factors will reduce schedule further

Cost Model Example: Systems Engineering (COSYSMO) Cost Drivers

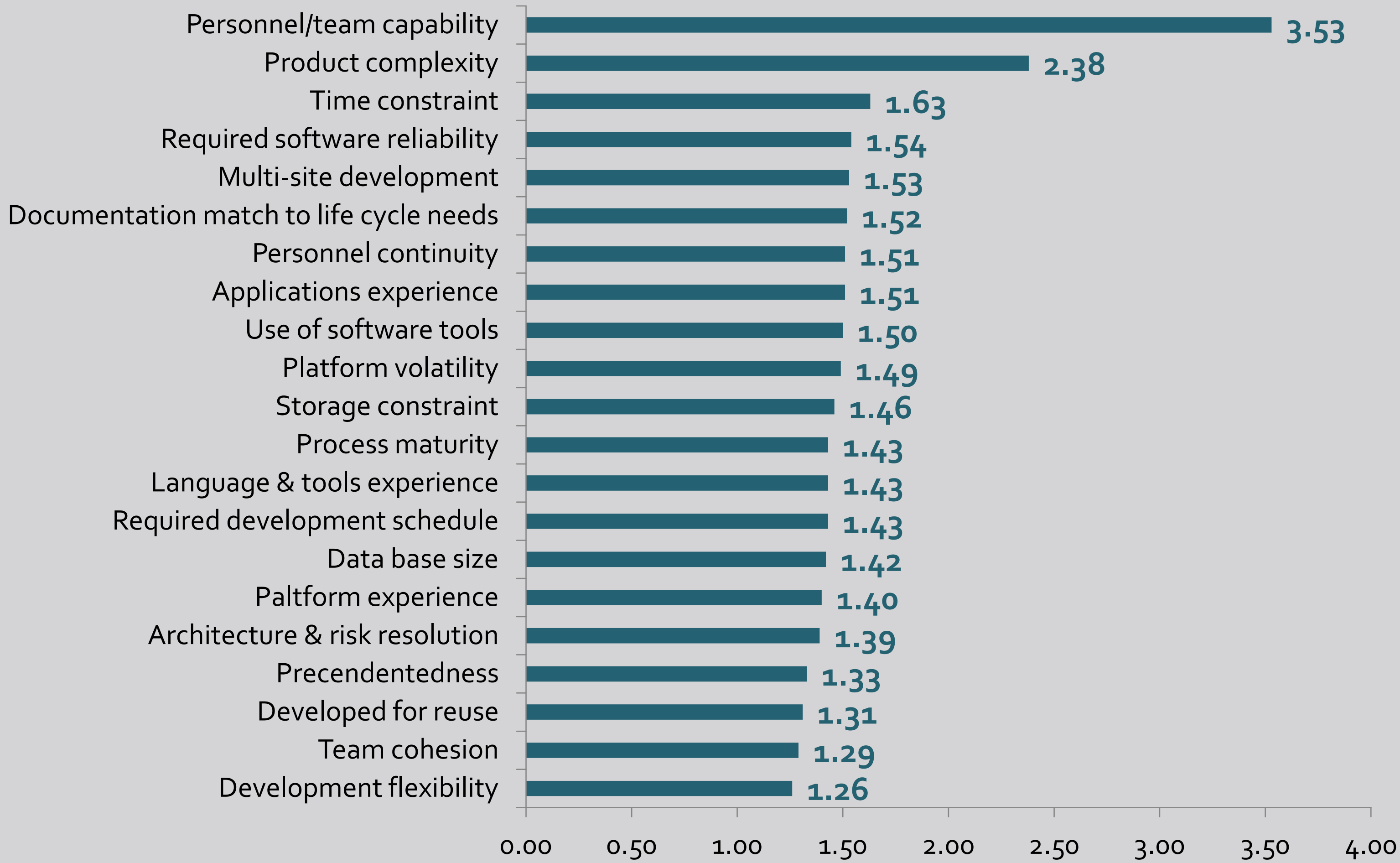


USC University of Southern California

USC CSSE

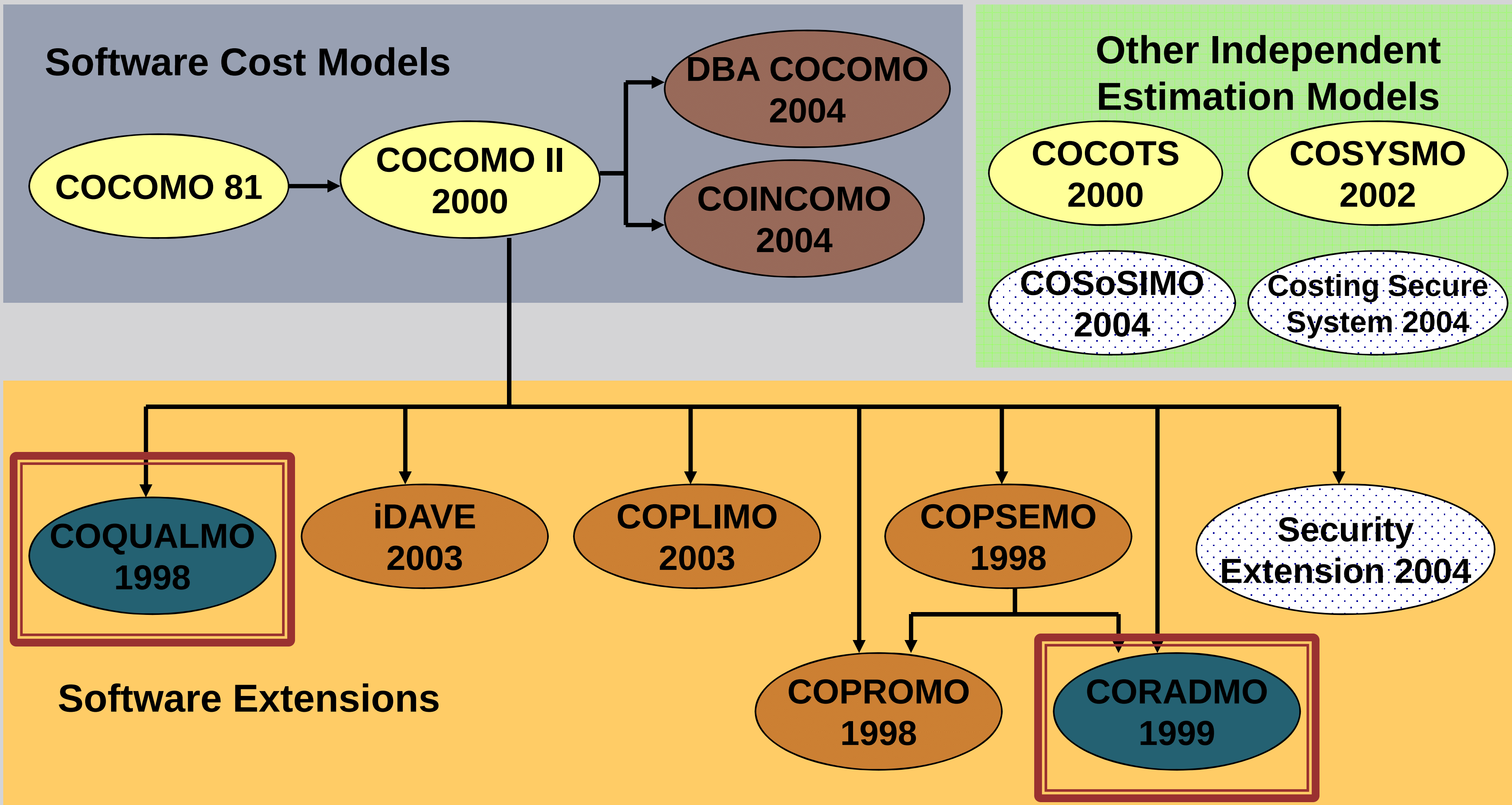
Cost Model Example: COCOMO Software Development Cost Factors

Software Development Productivity Range



USC University of Southern California

COCOMO Models to Support Tradespace Analyses



CORADMO-SE Schedule Drivers and Multipliers

Accelerators/Ratings	Very Low	Low	Nominal	High	Very High	Extra High
Product Factor: Multipliers	**1.09**	**1.05**	**1.0**	**0.96**	**0.92**	**0.87**
Simplicity	Extremely complex	Highly complex	Mod. complex	Moderately simple	Highly simple	Extremely simple
Element Reuse	None (0%)	Minimal (15%)	Some (30%)	Moderate (50%)	Considerate (70%)	Extensive (90%)
Low-Priority Deferrals	Never	Rarely	Sometimes	Often	Usually	Anytime
Models vs Documents	None (0%)	Minimal (15%)	Some (30%)	Moderate (50%)	Considerate (70%)	Extensive (90%)
Key Technology Maturity	>0 TRL 1,2 or >1 TRL 3	1 TRL 3 or > 1 TRL 4	1 TRL 4 or > 2 TRL 5	1-2 TRL 5 or >2 TRL 6	1-2 TRL 6	All > TRL 7
Process Factor: Multipliers	**1.09**	**1.05**	**1.0**	**0.96**	**0.92**	**0.87**
Concurrent Operational Concept, Requirements, Architecture, V&V	Highly sequential	Mostly sequential	2 artifacts mostly concurrent	3 artifacts mostly concurrent	All artifacts mostly concurrent	Fully concurrent
Process Streamlining	Heavily bureaucratic	Largely bureaucratic	Conservative bureaucratic	Moderate streamline	Mostly streamlined	Fully streamlined
General SE tool support CIM (Coverage, Integration, Maturity)	Simple tools, weak integration	Minimal CIM	Some CIM	Moderate CIM	Considerable CIM	Extensive CIM
Project Factors: Multipliers	**1.08**	**1.04**	**1.0**	**0.96**	**0.93**	**0.9**
Project size (peak # of personnel)	Over 300	Over 100	Over 30	Over 10	Over 3	≤ 3
Collaboration support	Globally distributed weak comm. , data sharing	Nationally distributed, some sharing	Regionally distributed, moderate sharing	Metro-area distributed, good sharing	Simple campus, strong sharing	Largely collocated, Very strong sharing
Single-domain MMPTs (Models, Methods, Processes, Tools)	Simple MMPTS, weak integration	Minimal CIM	Some CIM	Moderate CIM	Considerable CIM	Extensive CIM
Multi-domain MMPTs	Simple; weak integration	Minimal CIM	Some CIM or not needed	Moderate CIM	Considerable CIM	Extensive CIM
People Factors: Multipliers	**1.13**	**1.06**	**1.0**	**0.94**	**0.89**	**0.84**
General SE KSAs (Knowledge, Skills, Agility)	Weak KSAs	Some KSAs	Moderate KSAs	Good KSAs	Strong KSAs	Very strong KSAs
Single-Domain KSAs	Weak	Some	Moderate	Good	Strong	Very strong
Multi-Domain KSAs	Weak	Some	Moderate or not needed	Good	Strong	Very strong
Team Compatibility	Very difficult interactions	Some difficult interactions	Basically cooperative interactions	Largely cooperative	Highly cooperative	Seamless interactions
Risk Acceptance Factor: Multipliers	**1.13**	**1.06**	**1.0**	**0.94**	**0.89**	**0.84**
	Highly risk-averse	Partly risk-averse	Balanced risk aversion, accept	Moderately risk-accepting	Considerably risk-accepting	Strongly risk-accepting

USC University of Southern California

USC CSSE

Case Study

- A company division
- Diversified company defense applications
- Teams of roughly 20 SEs
- A sequential waterfall or Vee model in defining OpCons and requirements
- Then developing a system architecture that satisfies the requirements.
- Defense needs for more rapid SE

USC University of Southern California

USC CSSE

Product Factor
$1.09*1.09*1.05 = 1.25$

Product Factor
$1.25*0.92 = 1.15$

Process Factor
$1.09*1.05*0.96 = 1.1$

Project Factor
$0.96*0.96*0.96*1.04 = 0.92$

People Factor
$0.94*0.94*1.06 = 0.94$

Overall Factor
$1.15*1.1*0.92*0.94 = 1.09$

Accelerators/Ratings	Very Low	Low	Nominal			
Product Factor: Multipliers	1.09	1.05	1.0			
Simplicity	Extremely complex	Highly complex	Mod. complex	M		
Element Reuse	None (0%)	Minimal (15%)	Some (30%)	Moderate (50%)	Consider (70%)	Extensive (90%)
Low-Priority Deferrals	Never	Rarely	Sometimes			
Models vs Documents	None (0%)	Minimal (15%)	Some (30%)	M		
Key Technology Maturity	>0 TRL 1,2 or >1 TRL 3	1 TRL 3 or > 1 TRL 4	1 TRL 4 or > 2 TRL 5	1-2		
Process Factor: Multipliers	1.09	1.05	1.0			
Concurrent Operational Concept, Requirements, Architecture, V&V	Highly sequential	Mostly sequential	2 artifacts mostly concurrent	3		
Process Streamlining	Heavily bureaucratic	Largely bureaucratic	Conservative bureaucratic	Moderate streamline	Mostly streamlined	Fully streamlined
General SE tool support CIM (Coverage, Integration, Maturity)	Simple tools, weak integration	Minimal CIM	Some CIM			
Project Factors: Multipliers	1.08	1.04	1.0			
Project size (peak # of personnel)	Over 300	Over 100	Over 30			
Collaboration support	Globally distributed weak comm. , data sharing	Nationally distributed, some sharing	Regionally distributed, moderate sharing	Met		
Single-domain MMPTs (Models, Methods, Processes, Tools)	Simple MMPTS, weak integration	Minimal CIM	Some CIM			
Multi-domain MMPTs	Simple; weak integration	Minimal CIM	Some CIM or not needed			
People Factors: Multipliers	1.13	1.06	1.0			
General SE KSAs (Knowledge, Skills, Agility)	Weak KSAs	Some KSAs	Moderate KSAs			
Single-Domain KSAs	Weak	Some	Moderate	Good	Strong	Very strong
Multi-Domain KSAs	Weak	Some	Moderate or not needed	Good	Strong	Very strong
Team Compatibility	Very difficult interactions	Some difficult interactions	Basically cooperative interactions	Largely cooperative	Highly cooperative	Seamless interactions
Risk Acceptance Factor: Multipliers	1.13	1.06	1.0	0.94	0.89	0.84
	Highly risk-averse	Partly risk-averse	Balanced risk aversion, accept	Moderately risk-accepting	Considerably risk-accepting	Strongly risk-accepting

Case Study
- find a way to expedite

- Consider concurrent agile process approach
  - Sequential to 3 artifacts concurrently

Process Factor: Multipliers	1.09	1.05	1.0	0.96	0.92	0.87
Concurrent Operational Concept, Requirements, Architecture, V&V	Highly sequential	Mostly sequential	2 artifacts mostly concurrent	3 artifacts mostly concurrent	All artifacts mostly concurrent	Fully concurrent
Process Streamlining	Heavily bureaucratic	Largely bureaucratic	Conservative bureaucratic	Moderate streamline	Mostly streamlined	Fully streamlined
General SE tool support CIM (Coverage, Integration, Maturity)	Simple tools, weak integration	Minimal CIM	Some CIM	Moderate CIM	Considerable CIM	Extensive CIM

  - 1.09*1.05*0.96 = 1.1
  - 0.96*1.05*0.96 =0.96
  - Improve 0.96/1.1 = 0.88

Case Study - found several flaws

- With agile process, slow down 15%
- Use CORADMO-SE to analyze the influential factors;
  - transition to agile has several flaws

Factor	Scale	% change	Rationale
Key Technology Maturity	VH → N	1.0/0.92 = 1.09	Commit to immature solution Extra work and delays
General SE tool support	H → N	1.0/0.96 = 1.04	Using a mix of agile SE tools and their traditional SE tools made their SE tools less integrated
General SE KSAs	H → L	1.0/0.94 = 1.06	Still coming up the learning curve in their agile-SE KSAs
Team Compatibility	H → L	1.0/0.94 = 1.06	Management personnel continued to use traditional approaches

- Net Slow down factor - 0.88*1.09*1.04*1.06*1.06 = 1.13

Case Study
- to eliminate the slow down factors

- Use CORADMO-SE to identify improvements
- Initiatives
  - Concurrent V&V along with concurrent OpCons, Requirements, and architecture
    - 0.92/0.96= 0.97
  - Improve bureaucratic internal and external project and business process by streamlining
    - 0.96/1.04 = 0.92

Schedule Profile: 1.09 → 1.45 (Agile-not-ready effect) → 0.835 (CORADMO initiatives)

COQUALMO

- The COnstructive QUALity MOdel (COQUALMO)
- A set of combined cost, schedule and defect models enable tradeoffs between expedition, technical debt, and flexibility
- COQUALMO enables what-if analyses that demonstrate these impact

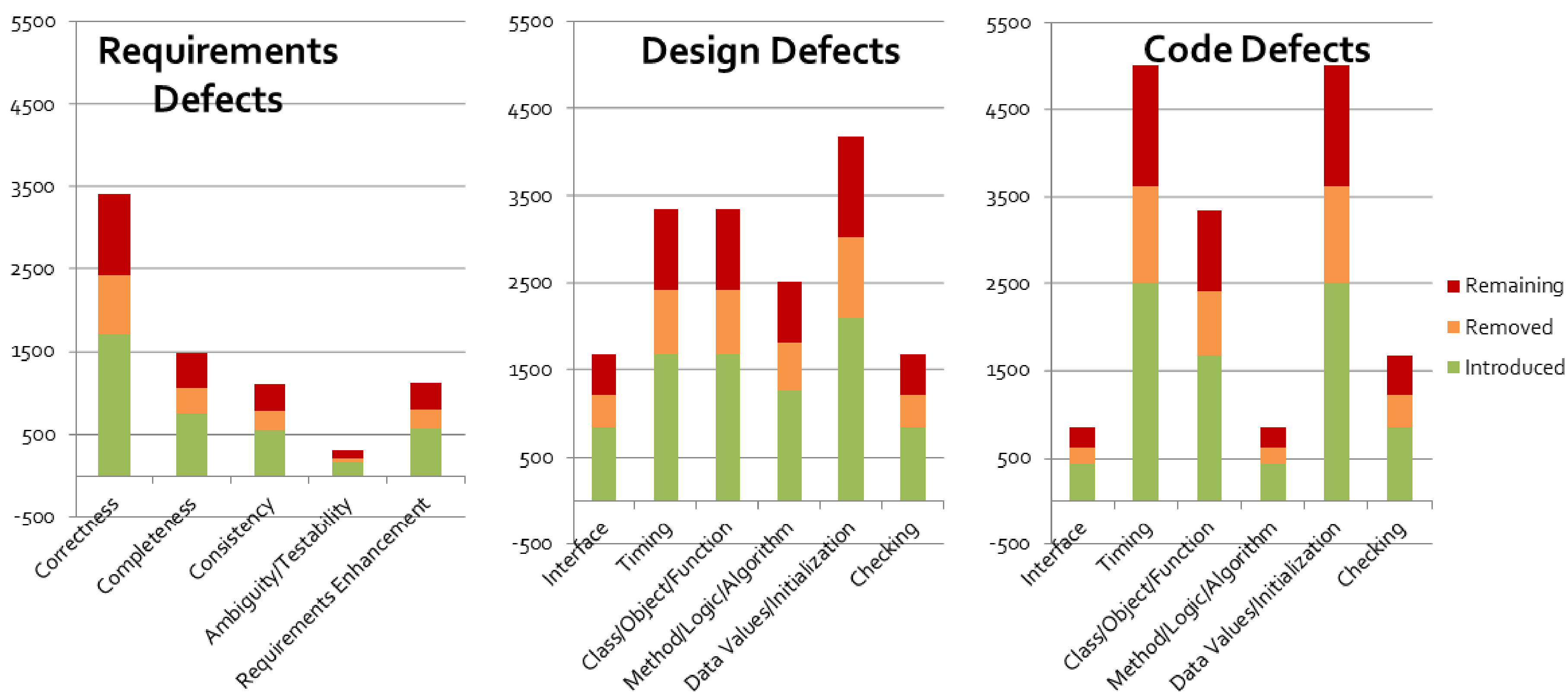
USC University of Southern California

USC CSSE

Expedited Project with not all risk mitigated, short schedule, not design for reuse and minimum defect removal activities

Center for Systems and Software Engineering

USC Viterbi School of Engineering
Celebrating 10^2 Years



Automated Analysis L Peer Reviews L Execution Testing and Tools L

Effort 1734.4 Person-Months

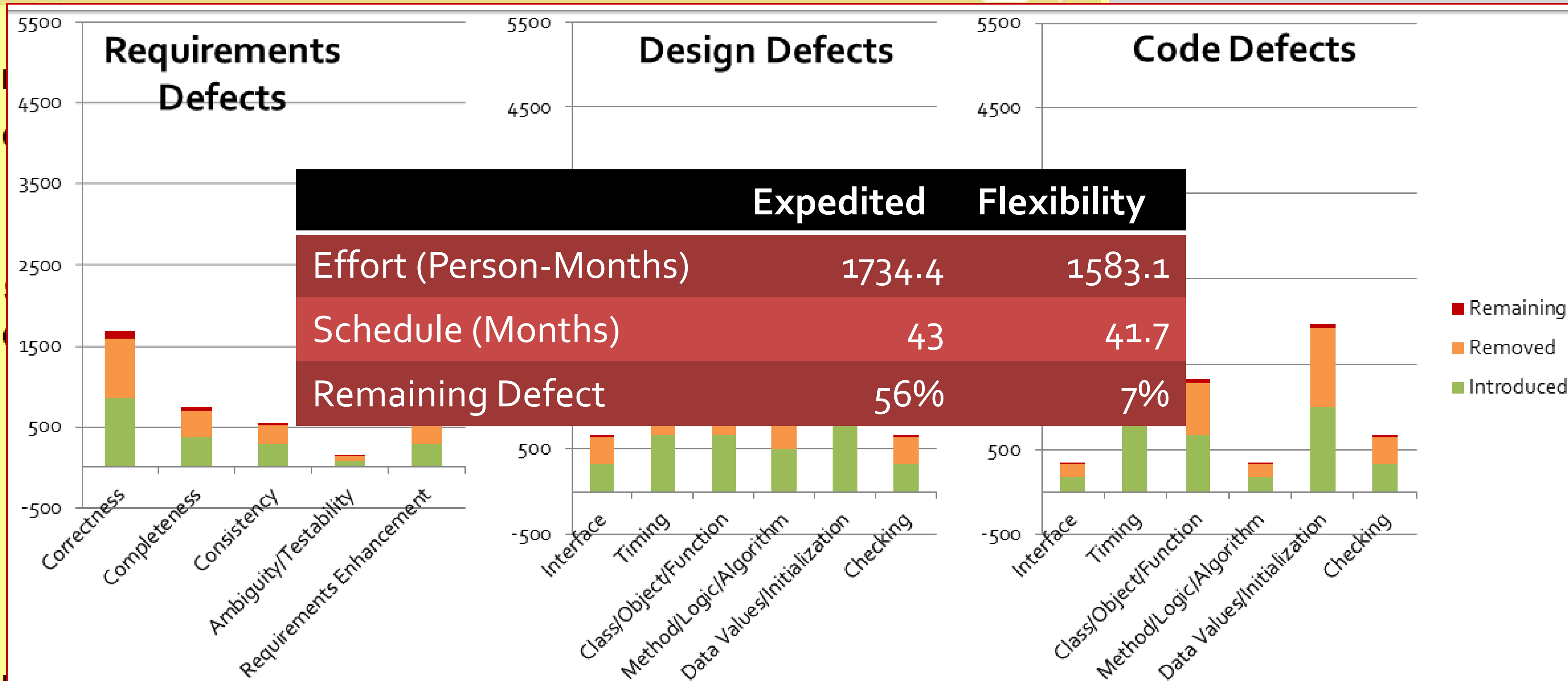
Schedule 43.0 Months

USC University of Southern California

Center for Systems and Software Engineering

USC Viterbi School of Engineering

Flexibility Project with very high risk mitigated, High reuse, long schedule and extensive defect removal activities



	Expedited	Flexibility
Effort (Person-Months)	1734.4	1583.1
Schedule (Months)	43	41.7
Remaining Defect	56%	7%

Automated Analysis VH | Peer Reviews VH | Execution Testing and Tools VH

Effort 1583.1 Person-Months

Schedule 41.7 Months

USC University of Southern California

USC CSSE

Conclusions

- System of systems engineering is lean to start with and builds upon tools, techniques, and approaches used for single systems
  - Expediting engineering
  - Valuing flexibility
  - Managing technical debt
- Key aspects to focus on for SoS
  - Interoperability of systems
    - Convergent protocols
    - Data standards
  - Migration of de facto architectures to more robust foundations
  - Coordination of schedules
  - Ability to take advantage of cross-cutting opportunities
- Beware of shortcuts to expediting… decreases in schedule that depend upon removal or minimization of quality activities may actual increase technical debt, effort and schedule

USC University of Southern California

USC CSSE

Acknowledgement

This material is based upon work supported in part by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

Discussion

USC University of Southern California

USC CSSE

Additional Information

USC University of Southern California

USC CSSE

Definitions

Capability

- The ability to achieve a desired effect under specified standards and conditions through combinations of means and ways across doctrine, organization, training, materiel, leadership and education, personnel, and facilities (DOTMLPF) to perform a set of tasks to execute a specified course of action. (*CJCSI 3170.01G*)

Technical Debt

- Delayed technical work or rework that is incurred when shortcuts are taken (*Ward Cunningham*)

Expedited Engineering

- General: Engineering techniques used to speed up the delivery of system capabilities
- Joint Urgent Operational Need (JUON): an urgent operation need identified by a combatant commander involved in an ongoing named operation. A JUON's main purpose is to identify and subsequently gain Joint Staff validation and resourcing of a solution, usually within days or weeks, to meet a specific high-priority combatant commander need. The scope of a combatant commander JUON will be limited to addressing urgent operational needs that: (1) fall outside of the established Service processes; and (2) most importantly, if not addressed immediately, will seriously endanger personnel or pose a major threat to ongoing operations. They should not involve the development of a new technology or capability; however, the acceleration of an Advanced Concept Technology Demonstration or minor modification of an existing system to adapt to a new or similar mission is within the scope of the JUON validation and resourcing process. (https://acc.dau.mil/CommunityBrowser.aspx?id=204169)

System Flexibility

- **Flexibility** is used as an attribute of various types of systems. In the field of engineering systems design, it refers to designs that can adapt when external changes occur. Flexibility has been defined differently in many fields of engineering, architecture, biology, economics, etc. In the context of engineering design one can define flexibility as the ability of a system to respond to potential internal or external changes affecting its value delivery, in a timely and cost-effective manner. Thus, flexibility for an engineering system is the ease with which the system can respond to uncertainty in a manner to sustain or increase its value delivery. Uncertainty is a key element in the definition of flexibility. Uncertainty can create both risks and opportunities in a system, and it is with the existence of uncertainty that flexibility becomes valuable. (http://en.wikipedia.org/wiki/Flexibility_(engineering))

USC University of Southern California

USC CSSE

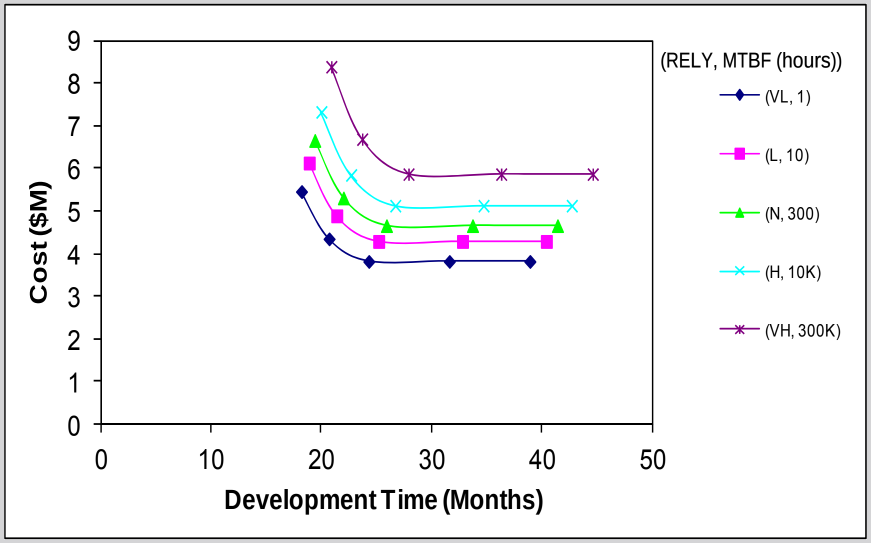
SoSE Core Element Description

- Translating Capability Objectives
  - Starts with an SoS need or new capability
  - Works to understand new capability and alternatives for providing it
- Understanding Systems and Their Relationships
  - Collects and maintains information about current state of the SoS and its CSs
- Assessing Performance to Capability Objectives
  - Evaluation of current performance and how performance meets current and future needs
- Developing/Evolving SoS Architecture
  - Evaluation of existing SoS architecture and identification of alternatives to mitigate limitations and improve performance
- Monitoring and Assessing Changes
  - Monitoring of CS non-SoS changes
- Addressing Requirements and Solution Options
  - Evaluation/prioritization of SoS requirements
  - Evaluation of solution options and selection of option
- Orchestrating Upgrades
  - Oversight activity to monitor progress of the CS SoS capability upgrades and mitigate obstacles

"ility" Tradespace Research to Date

- "ility" considerations
  - Include
    - New systems
    - Existing systems
    - Systems of systems
  - Start with balancing capabilities and their performance characteristics
  - Continue with investments in foundations and architecture
    - Manufacturability
    - Maintainability
    - Future options and opportunities
    - Total cost of ownership considerations
  - Plan for the retirement of aging, fragile systems that are difficult to maintain

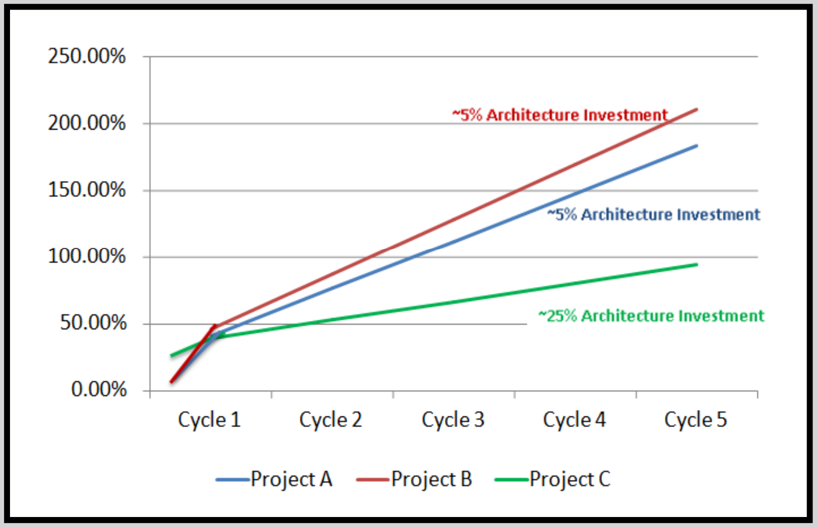
- Candidate models and tools to support affordability trades
  - "How much is enough"



"ility" Tradespace Research to Date

- "ility" considerations
  - Include
    - New systems
    - Existing systems
    - Systems of systems
  - Start with balancing capabilities and their performance characteristics
  - Continue with investments in foundations and architecture
    - Manufacturability
    - Maintainability
    - Future options and opportunities
    - Total cost of ownership considerations
  - Plan for the retirement of aging, fragile systems that are difficult to maintain

- Candidate models and tools to support affordability trades
  - "How much is enough"
  - Total cost of ownership



"ility" Tradespace Research to Date

- "ility" considerations
  - Include
    - New systems
    - Existing systems
    - Systems of systems
  - Start with balancing capabilities and their performance characteristics
  - Continue with investments in foundations and architecture
    - Manufacturability
    - Maintainability
    - Future options and opportunities
    - Total cost of ownership considerations
  - Plan for the retirement of aging, fragile systems that are difficult to maintain

- Candidate models and tools to support affordability trades
  - "How much is enough"
  - Total cost of ownership
  - System of systems engineering investments