Next Generation Trends:
Systems and Software Engineering

Barry Boehm, USC-CSSE
http://csse.usc.edu

CSSE Annual Research Review
April 15, 2015

Motivation

- "What helped me most in becoming a good hockey player was learning to skate to where the puck was going, rather than where it was or where it had been."
  - Wayne Gretzky, NHL Hall of Fame

- "Reflection in action," or asking, "How could we have done our last project better?" is actually skating to where the puck has been.
  - It is very valuable, but needs to be balanced with anticipating the future
  - For example, don't stop at CMMI Level 4

Outline

- **The Future of Systems and Software**
  - **Original 2005 presentation**
    - **8 surprise-free trends; 2 wild-card trends**
  - **Changes 2005-2010; 2010-2015**
  - **Systems and software engineering opportunities and challenges**

- **Conclusions: Continuous Adaptation**
  - **Research, acquisition, staffing/education**

The Future of Systems and Software: 2005

- **Eight surprise-free trends**
  1. Increasing integration of SysE and SwE
  2. User/Value focus
  3. Software Criticality and Dependability
  4. Rapid, Accelerating Change
  5. Distribution, Mobility, Interoperability, Globalization
  6. Complex Systems of Systems
  7. COTS, Open Source, Reuse, Legacy Integration
  8. Computational Plenty
- **Two wild-card trends**
  9. Autonomy Software
  10. Combinations of Biology and Computing

2010 Trends Largely Missed in 2005

- Nanotechnology megasensor-intensive smart systems
- Search and mining of ultralarge data aggregations
- Software implications of multicore chips
- Rapid growth of cloud computing, service-orientation
- Rapid growth of social networking technologies

The Future of Systems and Software: 2010

Eight surprise-free trends

1. Rapid, Accelerating Change
2. Software Criticality and Dependability
3. Complexity; Global/Mobile Systems of Systems
4. COTS, Open Source, Services, Legacy Integration
5. Smart Systems; Mining huge volumes of data
6. User Evolution and End Value Focus
7. Computational Plenty and Multicore Chips
8. Increasing integration of SysE and SwE

Two wild-card trends

9. Autonomy Software
10. Combinations of Biology and Computing

2015 Trends Largely Missed in 2010

- From Cyber-Physical Systems to Cyber-Physical-Human Systems
- Proliferation of Apps: Full Interoperability
- Agile Methods Meet Complex Trusted Systems
- Proliferation of Autonomous Systems
- Changes in Labor Force Supply and Demand
- Advanced Human Prosthetics

The Future of Systems and Software: 2015

1. Reliable Autonomy and Cyber-Physical-Human Systems
2. Rapid Change: Set-Based Design and Requirements
3. Balancing Multi-Stakeholder Objectives, Qualities
4. Complexity; Global/Mobile Systems of Systems
5. COTS, Open Source, Services, Legacy Integration
6. Smart Systems; Mining huge volumes of data
7. Advanced Human Prosthetics
8. Lifelong Education: Early STEM; T-Shaped Software Engineers

Reliable Autonomy and Human-Systems Integration

- **Balance human oversight of autonomous agents with agent-based oversight of human deficiencies**
  - **Autonomy failure modes: agent cooperation; agent spoofing; big-data spoofing; self-modifying systems; positive feedback stability**
  - **Need for multidiscipline-stakeholder collaboration support**
  - **Need integrated modeling of human-cyber-physical roles, performance**

- **HSI methods, processes and tools. Growing need with more complex systems, especially autonomy.**
  - **What is the tradespace analysis associated with human vs. machine executive control of system capabilities and how is it determined?**

Trusted Systems: Systemic Assurance

- **Modeling, analyzing, and assuring semi-autonomous and autonomous systems**
  - **How to model and validate autonomy requirements?**
    - **Rules of engagement**
    - **Resilient and adaptive responses**
    - **Human roles and interventions**

- **Managing traceability at full scale: requirements, architecture, design, models, implementation, tests, ops logs, etc.**
  - **Can argumentation structures be engineered to achieve scale?**
    - **How to validate argumentation structures?**
    - **How to link argumentation structures with proof structures, models, analytics?**

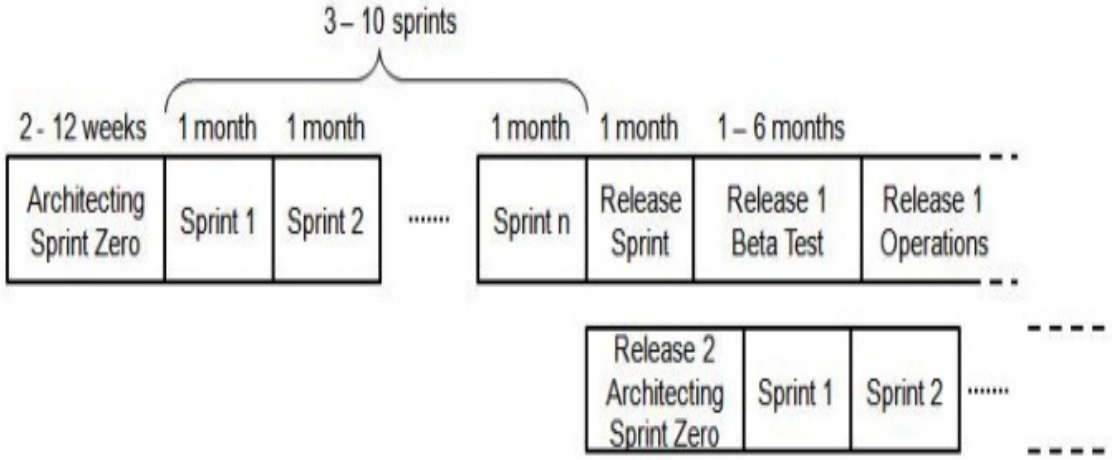
- **Systems engineering data**
  - **What is the nature of the tooling required to manage the "big data" of a full SE process?**
    - **Kinds of data (what's in the "*SEEN*" – the *SE Engineering Notebook*): informal, formal, traceability, models, etc.**
  - **How to provide effective "analytic visibility" to Program Office personnel?**

- **Open source in systems**
  - **What kinds of evidence need to be produced to support confident adoptions?**
    - **The full access – and the ability to insert (as a stakeholder) any additional quality steps into the build – creates benefit that could counter the negatives (loss of control, visibility to adversaries).**

- **Curriculum**
  - **How can SE principles be effectively inserted into existing software engineering and ECE curricula?**
    - **This could broaden the population of students exposed to the techniques and tools of SE.**

Architected Agile Approach

- **Uses Scrum of Scrums approach**
  - **Up to 10 Scrum teams of 10 people each**
  - **Has worked for distributed international teams**
  - **Going to three levels generally infeasible**
- **General approach shown below**
  - **Often tailored to special circumstances**



Set-based Design & Rqts.

- Anticipate future directions of change
  - Areas of uncertainty in requirements
    - Long-lead items, low TRLs, independently-evolving SoS elements
  - Previous-systems change areas
  - Architect system around sources of change
  - Include mechanisms for early validation
  - Prioritize capabilities to keep within budget
- Create tradespace via ranges of quality requirements
  - Example: 1-second response time desired; 4 seconds acceptable
- Keep sets of options open as late as possible
  - Be responsive to emerging threats and technology opportunities
- Facilitated by tools to generate large numbers of options
  - And computer models to evaluate them on hundreds of parameters

Projects A and B Major Rework Sources

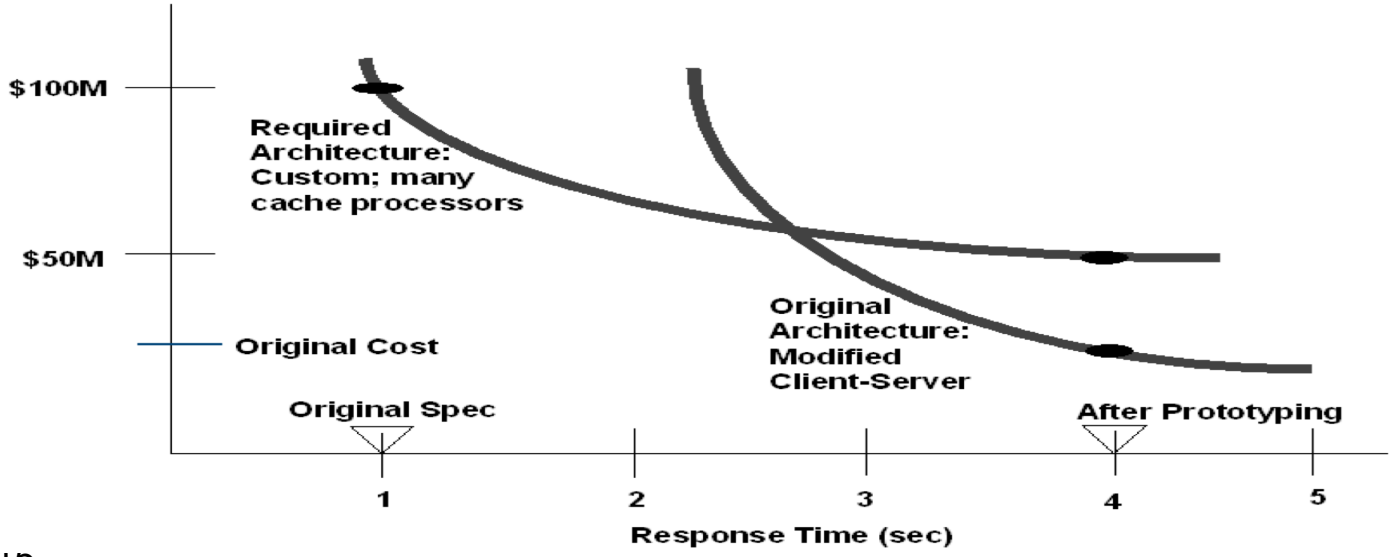
Change processing over 1 person-month = 152 person-hours

Category	Project A	Project B
Extra long messages		3404+626+443+328+244= 5045
Network failover	2050+470+360+160= 3040	
Hardware-software interface	620+200= 820	1629+513+289+232+166= 2832
Encryption algorithms		1247+368= 1615
Subcontractor interface	1100+760+200= 2060	
GUI revision	980+730+420+240+180 =2550	
Data compression algorithm		910
External applications interface	770+330+200+160= 1460	
COTS upgrades	540+380+190= 1110	741+302+221+197= 1461
Database restructure	690+480+310+210+170= 1860	
Routing algorithms		494+198= 692
Diagnostic aids	360	477+318+184= 979
TOTAL:	**13620**	**13531**

Importance of SQ Tradeoffs

Major source of DoD system overruns

- SQs have systemwide impact
  - System elements generally just have local impact
- SQs often exhibit asymptotic behavior
  - Watch out for the knee of the curve
- Best architecture is a discontinuous function of SQ level
  - "Build it quickly, tune or fix it later" highly risky
  - Large system example below



Commit with Confidence: Evidence-Based Decision Milestones

- *Evidence* provided by developer and validated by independent experts that:
- If the system is built within the specified architecture envelope, it will
  - Satisfy the specified operational concept and requirements
    - Capability, interfaces, level of service, and evolution
  - Be buildable by the developers within the budgets and schedules in the plan
  - Generate a viable return on investment in mission performance
  - Generate satisfactory outcomes for all of the success-critical stakeholders
- Shortfalls in evidence are uncertainties and risks
  - Should be resolved or covered by risk management plans
- Assessed in increasing detail at major decision milestones
  - Uncertainty-managed level of evidence detail
  - Serves as basis for stakeholders' commitment to proceed
  - Serves to synchronize and stabilize concurrently engineered elements
- Evidence for assurance links requirements, architecture, implementation, ops
  - Dynamic traceability greatly facilitates evolution with assurance

Can be used to strengthen current schedule- or event-based reviews

The Future of Systems and Software: 2015

1. Reliable Autonomy and Cyber-Physical-Human Systems
2. Rapid Change: Set-Based Design and Requirements
3. ⇨ Balancing Multi-Stakeholder Objectives, Qualities
4. Complexity; Global/Mobile Systems of Systems
5. COTS, Open Source, Services, Legacy Integration
6. Smart Systems; Mining huge volumes of data
7. Advanced Human Prosthetics
8. Lifelong Education: Early STEM; T-Shaped Software Engineers

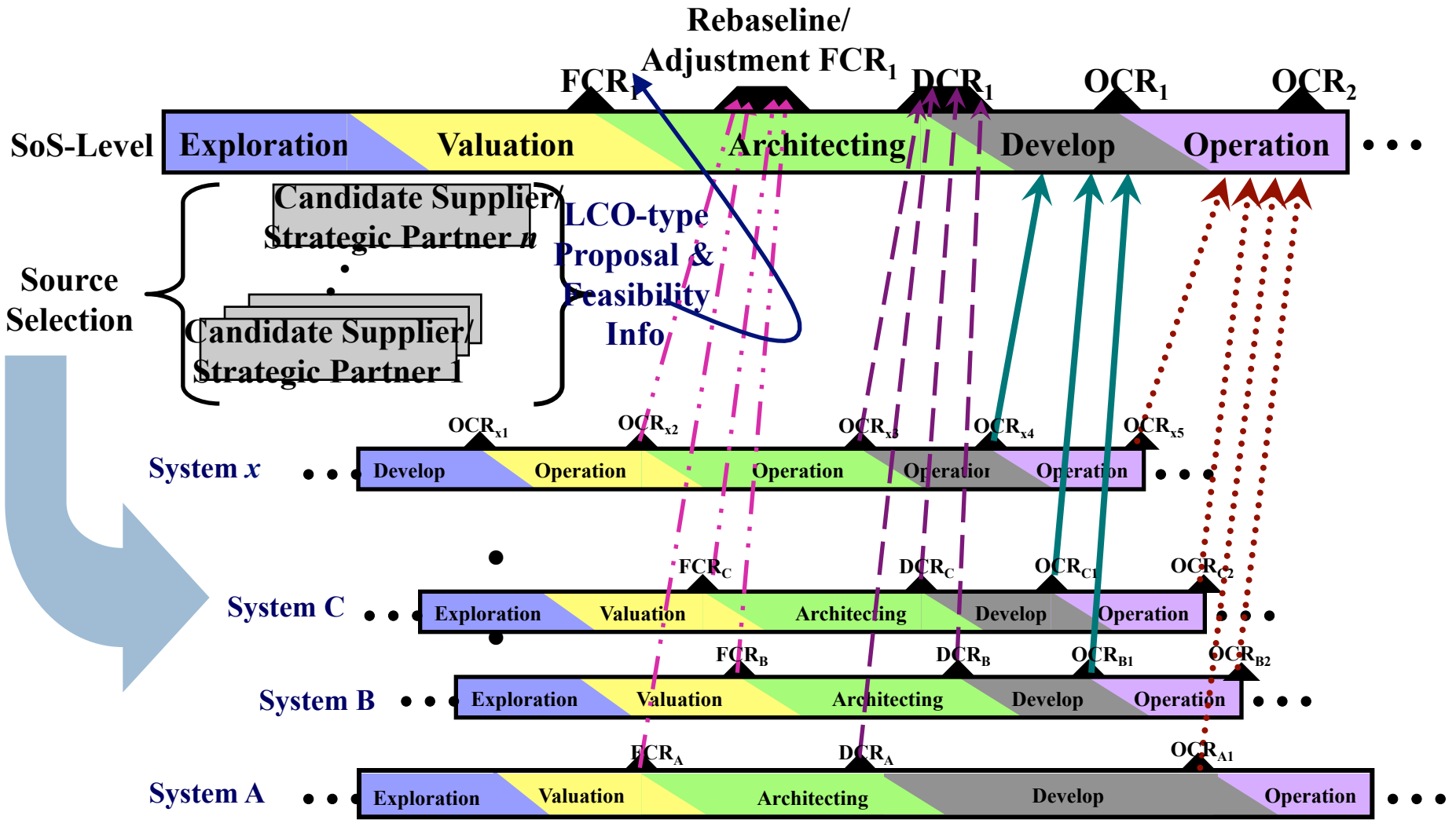
Example of SQ Value Conflicts: Security IPT

- **Single-agent key distribution; single data copy**
  - **Reliability: single points of failure**
- **Elaborate multilayer defense**
  - **Performance: 50% overhead; real-time deadline problems**
- **Elaborate authentication**
  - **Usability: delays, delegation problems; GUI complexity**
- **Everything at highest level**
  - **Modifiability: overly complex changes, recertification**

3. Complexity and Global Software-Intensive Systems of Systems (SISOS)

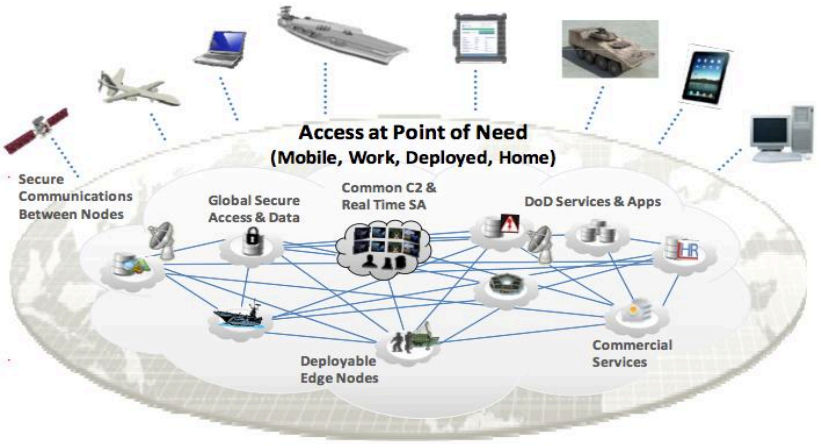
- **Lack of integration among stovepiped systems causes**
  - **Unacceptable delays in service**
  - **Uncoordinated and conflicting plans**
  - **Ineffective or dangerous decisions**
  - **Inability to cope with fast-moving events**
- **Increasing SISOS benefits**
  - **See first; understand first; act first**
  - **Network-centric operations coordination**
  - **Transformation of business/mission potential**
  - **Interoperability via Integrated Enterprise Architectures**

Future DoD Challenges: Systems of Systems



Enterprise and Systems of Systems Grand Challenge

Strategy:



- ***Model:*** **Develop MPTs that allow quick and insightful modeling of enterprises/SoSs so that the effects of changes in policies, practices, components, interfaces, and technologies can be anticipated and understood in advance of their implementation**
- ***Example Need: SoS Change Impact Analysis***

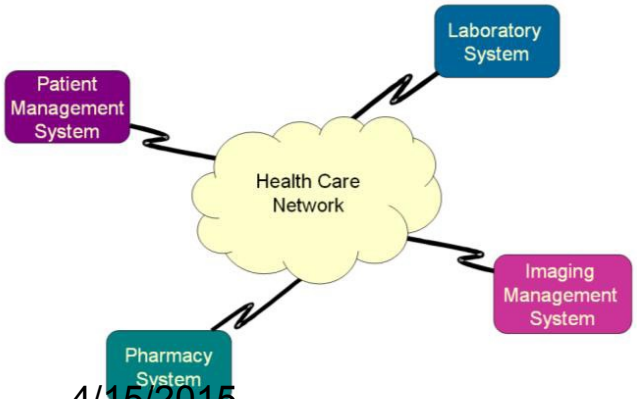
Acquire: **Develop MPTs that allow insight into enterprise/SoS acquisition approaches in the face of significant uncertainty and change to minimize unintended consequences and unforeseen risks**

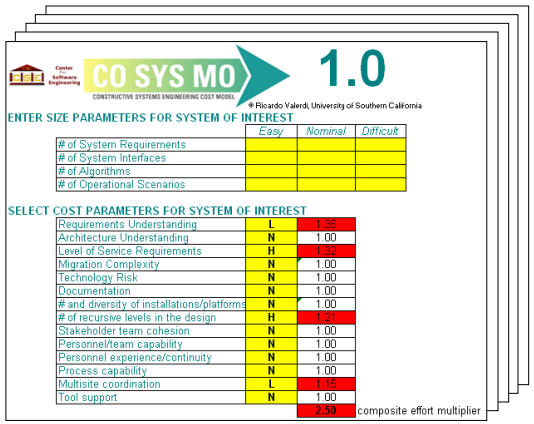
- ***Evolve:*** **Develop MPTs that facilitate evolving and growing an enterprise/SoS, including insight into different architectural and integration approaches that facilitate evolution in the face of uncertainty and change in how an enterprise/SoS is employed, the technologies available to realize it, and the environment in which it exists**

- ***Verify:*** **Develop MPTs that allow the properties of an enterprise/SoS to be anticipated, monitored and confirmed during development and evolution, including an enterprise/SoS which includes legacy systems that are in operation while development and evolution are underway**

GTRI-USC SysML Building Blocks for Cost Modeling

- Implemented reusable SysML building blocks
  - Based on SoS/COSYSMO SE cost (effort) modeling work by Lane, Valerdi, Boehm, et al.
- Successfully applied building blocks to healthcare SoS case study
- Provides key step towards affordability trade studies involving diverse "-ilities"





Aspect	Formula	Calculated Effort
SoSE effort (Equation 5)	Effort = $38.55*[((SoS_{CR} / SoS_{Treq})* (SoS_{Treq})^{1.06} * EM_{SOS\text{-}CR}) + ((SoS_{MR} / SoS_{Treq})* (SoS_{Treq})^{1.06} * EM_{SOS\text{-}MR}*OSF)] /152$ = $38.55*[((50 / 52) * (52)^{1.06} * 2.50) + (20/52)*(52)^{1.06}* 0.47 * 10\%)] / 152$	40.41
Pharmacy System effort (Equation 4)	Effort = $38.55*[(1.0+CS_{SoSmp}) * ((SoS_{CSalloc}/CS_{TreqSoSE})* (CS_{TreqSoSE})^{1.06}* EM_{CS\text{-}CRwSOSE}) + (CS_{nonSoS}/CS_{TreqSoSE}) * (CS_{TreqSoSE})^{1.06} * EM_{CSnonSOS}] /152$ = $38.55 *[(1.15) * ((50/70)*(70)^{1.06} * 1.06 + (20/70) * (70)^{1.06}*0.72] / 152$	22.02
Laboratory System effort (Equation 4)	Effort = $38.55*[(1.0+CS_{SoSmp}) * ((SoS_{CSalloc}/CS_{TreqSoSE})* (CS_{TreqSoSE})^{1.06}* EM_{CS\text{-}CRwSOSE}) + (CS_{nonSoS}/CS_{TreqSoSE}) * (CS_{TreqSoSE})^{1.06} * EM_{CSnonSOS}] /152$ = $38.55 *[(1.15) * ((50/50)*(50)^{1.06} * 1.06 + 0] / 152$	19.55
Imaging System effort (Equation 4)	Effort = $38.55*[(1.0+CS_{SoSmp}) * ((SoS_{CSalloc}/CS_{TreqSoSE})* (CS_{TreqSoSE})^{1.06}* EM_{CS\text{-}CRwSOSE}) + (CS_{nonSoS}/CS_{TreqSoSE}) * (CS_{TreqSoSE})^{1.06} * EM_{CSnonSOS}] /152$ = $38.55 *[(1.15) * ((50/50)*(50)^{1.06} * 1.06 + 0] / 152$	19.55
New infrastructure component effort (Equation 1)	Effort = $38.55*EM*(size)^{1.06}/152$ = $38.55 * 1.0 * (100)^{1.06} / 152$	33.43
	Total Effort:	134.96

The Future of Systems and Software: 2015

1. Reliable Autonomy and Cyber-Physical-Human Systems
2. Rapid Change: Set-Based Design and Requirements
3. Balancing Multi-Stakeholder Objectives, Qualities
4. Complexity; Global/Mobile Systems of Systems
5. COTS, Open Source, Services, Legacy Integration
6. Smart Systems; Mining huge volumes of data
7. Advanced Human Prosthetics
8. ⇨Lifelong Education: Early STEM; T-Shaped Software Engineers

SysE and SwE Education Implications

- **Current SysE and SwE students will be practicing into the 2050s. Their education should consider the following:**
  - **Early education: Systems thinking, STEM analysis and synthesis**
  - **Anticipating future trends and preparing students to deal with them;**
  - **Capitalizing on information technology to enable the delivery of just-in-time and web-based education;**
  - **Monitoring current principles and practices and separating timeless principles from outdated practices;**
  - **Participating in leading-edge software engineering research and practice and incorporating the results into the curriculum;**
  - **Packaging smaller-scale educational experiences in ways that apply to large-scale projects;**
  - **Helping students learn how to learn, through state-of-the-art analyses, future-oriented educational games and exercises, and participation in research; and**
  - **Offering lifelong learning opportunities for systems engineers who must update their skills to keep pace with the evolution of best practices**