



Software Architecture Challenges for Complex Systems of Systems

Barry Boehm, USC

GSAW 2003 Keynote

March 6, 2003

(boehm@sunset.usc.edu)

(<http://sunset.usc.edu>)



Complex Systems of Systems (CSOS): Software Benefits and Architecture Challenges

- **CSOS characteristics and software benefits**
- **Software benefits and architecture challenges**
- **Conclusions**



CSOS Characteristics and Software Benefits (relative to hardware)

<ul style="list-style-type: none">• Many component systems and contractors with wide variety of users and usage scenarios—including legacy systems	<ul style="list-style-type: none">• Ease of accommodating many combinations of options• Ease of tailoring various system and CSOS versions
<ul style="list-style-type: none">• Need to rapidly accommodate frequent changes in missions, environment, technology, and interoperating systems	<ul style="list-style-type: none">• Rapidly adaptable• Rapidly upgradeable• Near-free COTS technology upgrades
<ul style="list-style-type: none">• Need for early capabilities	<ul style="list-style-type: none">• Flexibility to accommodate concurrent and incremental development



CSOS Software Benefits and Architecture Challenges

- **Accommodating many combinations of options**
 - **Development speed; integration; cross-system KPP's**
- **Accommodating many combinations of systems and contractors**
 - **Subcontractor specifications, incompatibilities, change management**
- **Rapid tailoring and upgrade of many combinations of options**
 - **Version control and synchronous upgrade propagation**
- **Flexibility, rapid adaptability, incremental development**
 - **Subcontractor chain increment synchronization; requirements and architecture volatility**
- **Near-free COTS technology upgrades**
 - **COTS upgrade synchronization; obsolescence; subcontractor COTS management**

Many CSOS Options and Software Development Speed

- **Risk #1: Limited speed of CSOS Software Development**
 - Many CSOS scenarios require close coupling of complex software across several systems and subsystems
 - Well-calibrated software estimation models agree that there are limits to development speed in such situations
 - Estimated development schedule in months for closely coupled SW with size measured in equivalent KSLOC (thousands of source lines of code):

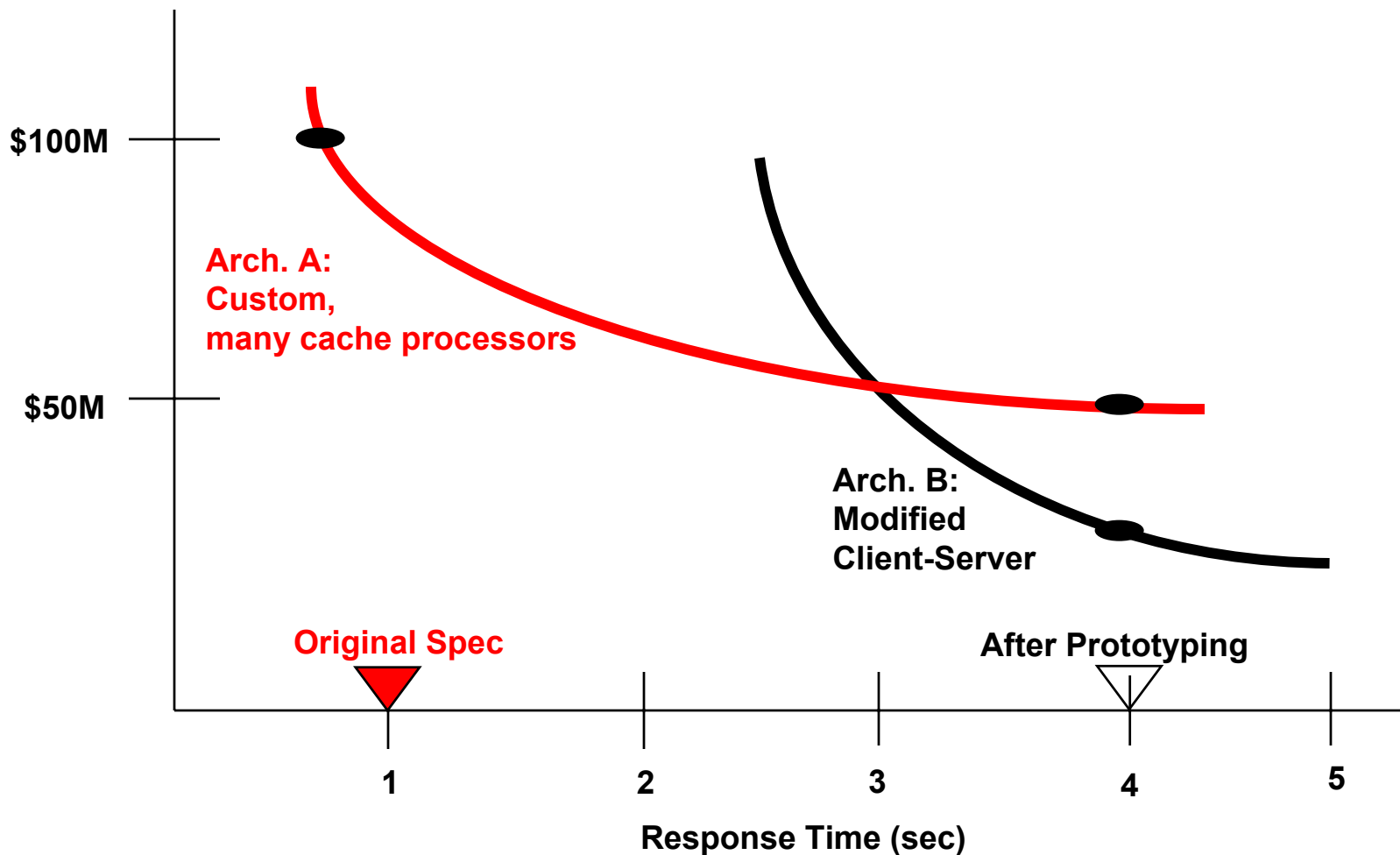
$$\text{Months} \approx 5 * \sqrt[3]{\text{KSLOC}}$$

- KSLOC	300	1000	3000	10,000
- Months	33	50	72	108

Architecture Challenge: Limited Speed of CSOS Software Development

- **Architecture Strategy #1: Architect the CSOS software to be able to develop independent software units in parallel and have them cleanly integrate at the end.**
- **Architecture Challenges**
 - **Key Performance Parameters (KPP's) and architecture discontinuities**
 - **Many subcontractors: How soon to define subcontractor interfaces**

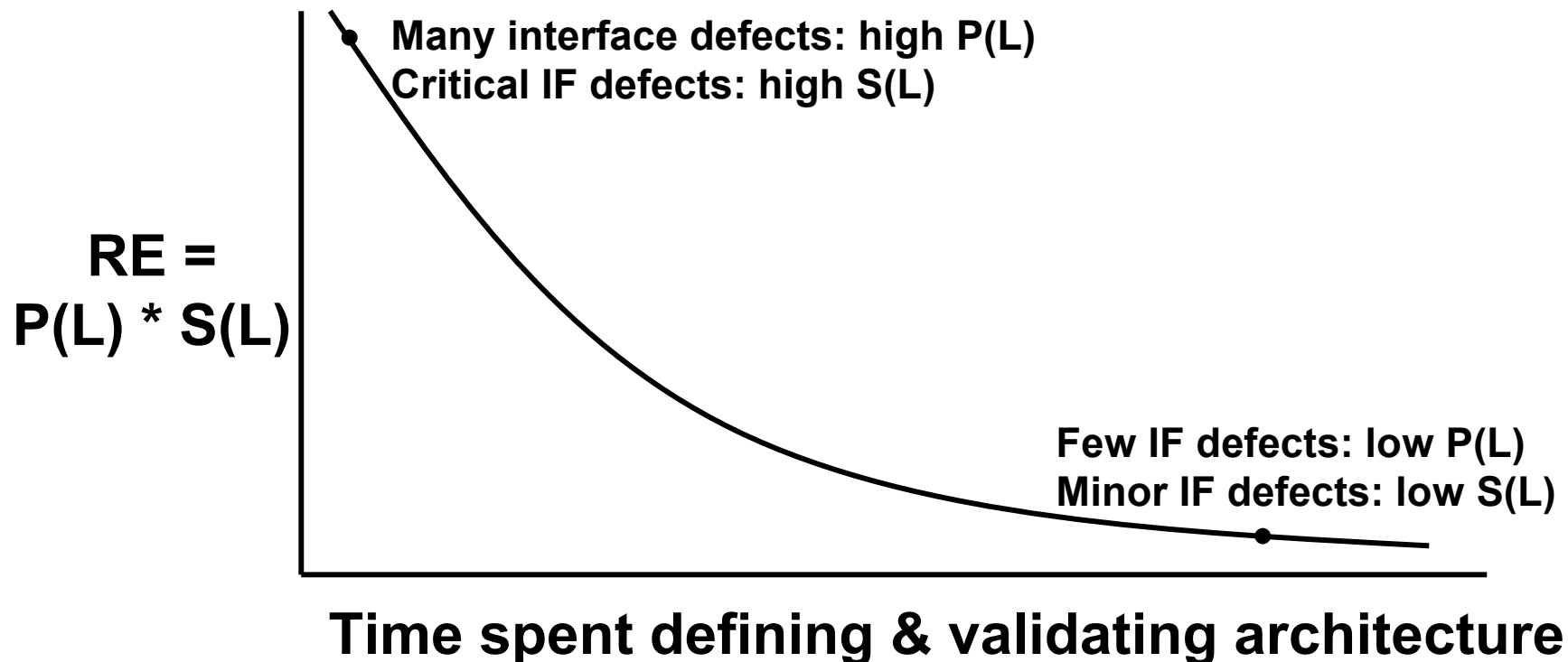
Best Software Architecture Is A Discontinuous Function of KPP Level



How Soon to Define Subcontractor Interfaces?

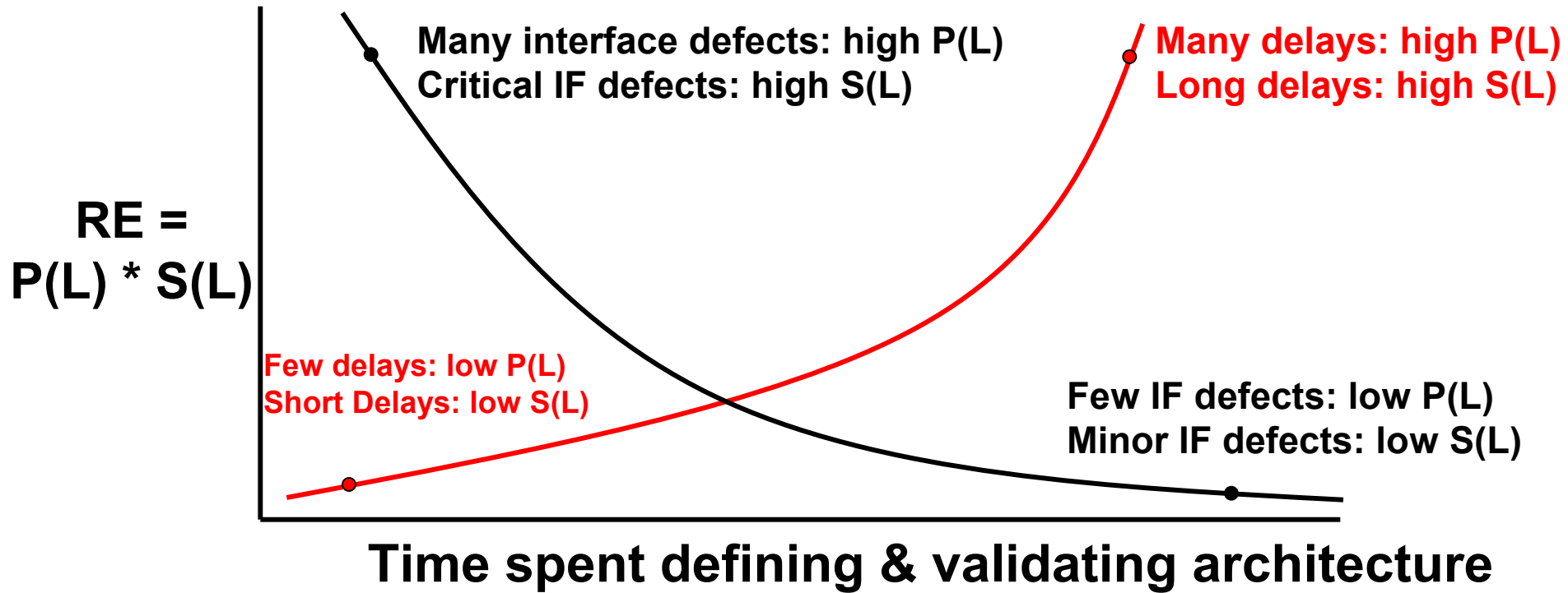
Risk exposure $RE = \text{Prob}(\text{Loss}) * \text{Size}(\text{Loss})$

-Loss due to rework delays



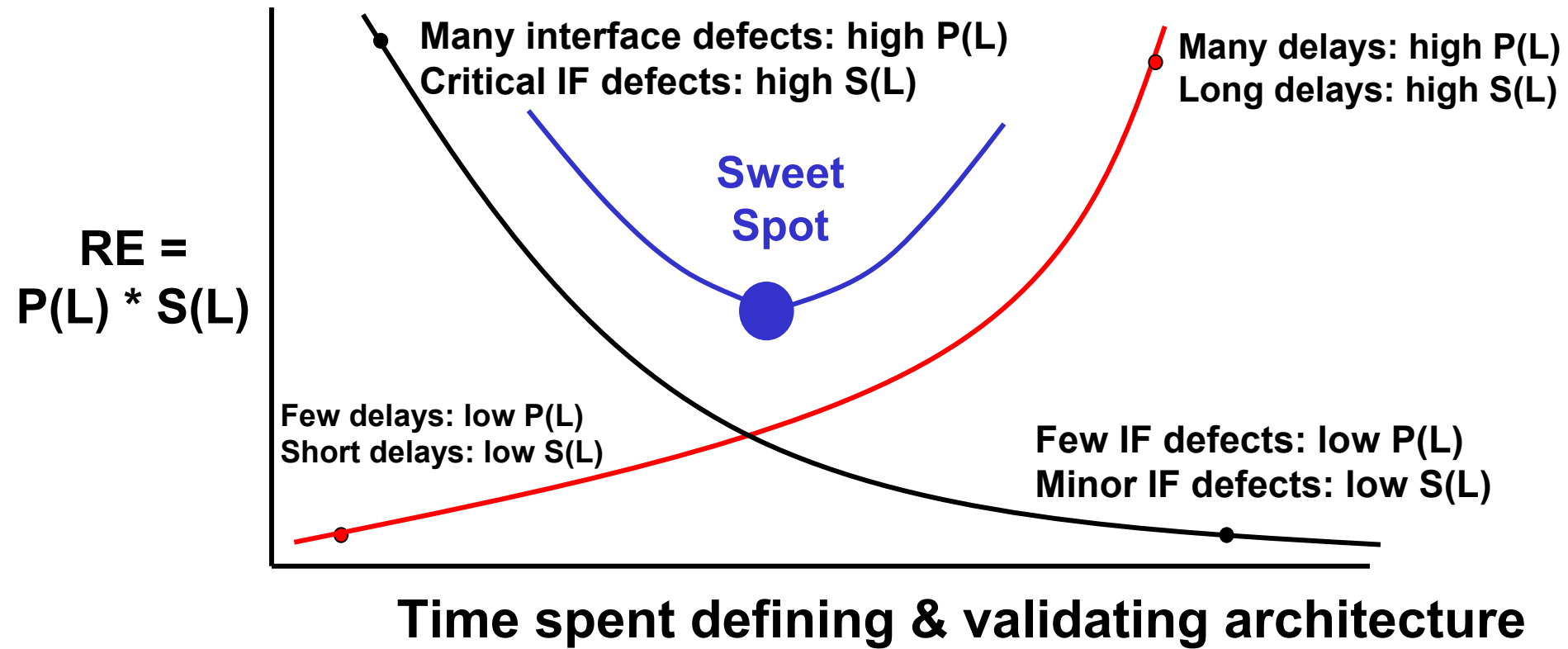
How Soon to Define Subcontractor Interfaces?

- Loss due to rework delays
- **Loss due to late subcontract startups**



How Soon to Define Subcontractor Interfaces?

- Sum of Risk Exposures



How Soon to Define Subcontractor Interfaces?

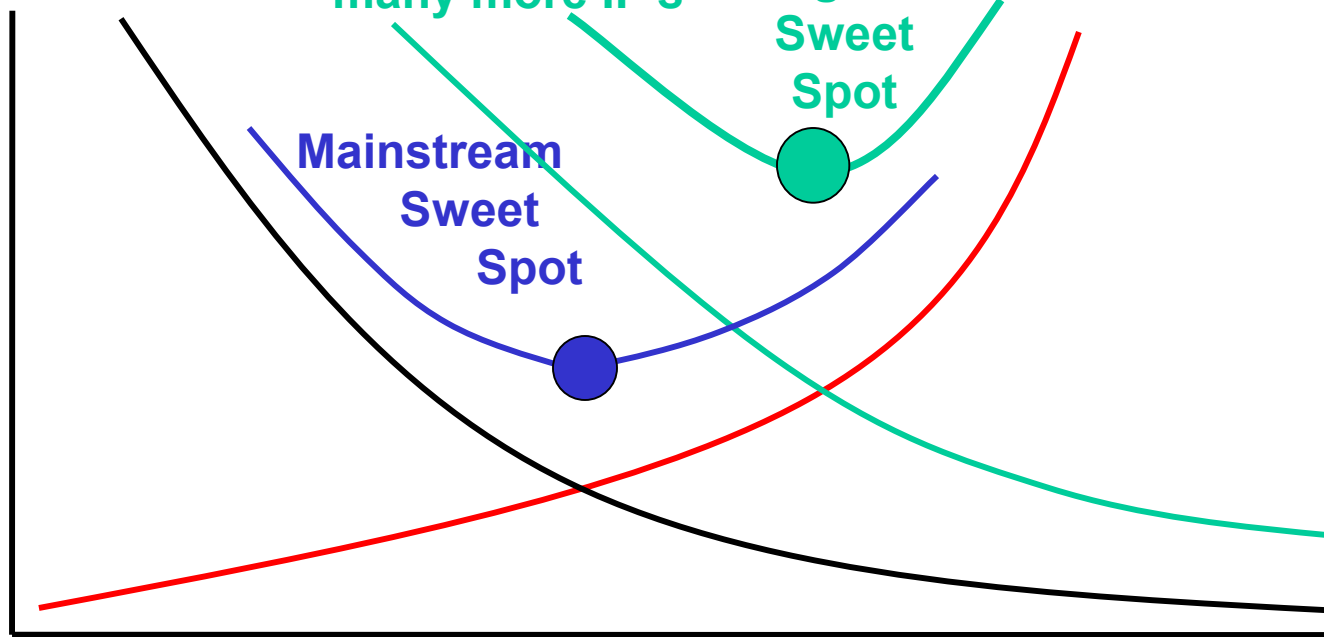
-Very Many Subcontractors

Higher $P(L)$, $S(L)$:
many more IF's

High-Q
Sweet
Spot

Mainstream
Sweet
Spot

$$RE = P(L) * S(L)$$



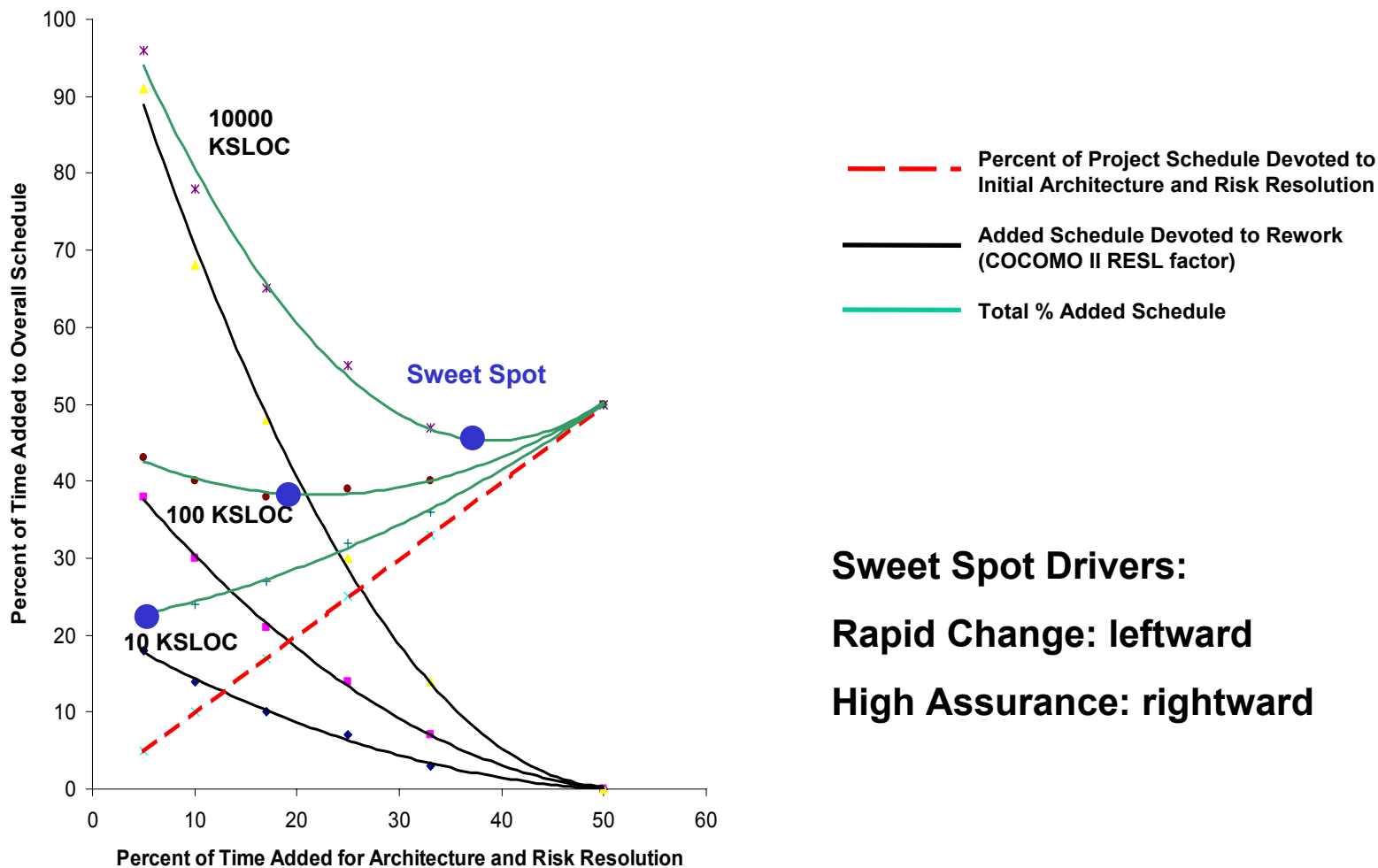
Time spent defining & validating architecture

Risk #4: Delayed CSOS availability due to many-subcontractor IF rework

Strategy #4: Invest more time in architecture definition

How Much Architecting Is Enough?

-A COCOMO II Analysis



Sweet Spot Drivers:

Rapid Change: leftward

High Assurance: rightward

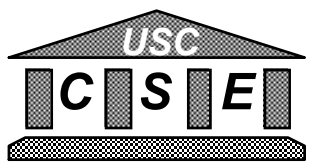
Rapid, Synchronous Software Upgrades

- **Risk #2. Out-of-synchronization software upgrades will be a major source of operational losses**
 - Software crashes, communication node outages, out-of-synch data, mistaken decisions
 - Extremely difficult to synchronize multi-version, distributed, mobile-platform software upgrades
 - Especially if continuous-operation upgrades needed
- **Strategy #2a. Architect software to accommodate continuous-operation, synchronous upgrades**
 - E.g., parallel operation of old and new releases while validating synchronous upgrade
- **Strategy #2b. Develop operational procedures for synchronous upgrades in software support plans**
- **Strategy #2c. Validate synchronous upgrade achievement in operational test & evaluation**



Rapid Adaptability to Change: Evolving Software Architecture

- **Risk #3.** Software architecture will need to change & adapt to rapidly changing priorities and architecture drivers
 - new COTS releases;
 - evolving enterprise standards and policies;
 - emerging technologies and competitor threats
- **Strategy #3a.** Empower a focal-point integrator of the software architecture and owner of the critical software infrastructure.
- **Strategy #3b.** Raise the organizational level of the owner of the software architecture to a very high level in the CSOS organizational structure.



Rapid Adaptability to Change: Architecture Evolution and Subcontracting

- **Risk #4. The CSOS software architecture will inevitably change. Inflexible subcontracting will be a major source of delays and shortfalls.**
- **Strategy #4. Develop subcontract provisions enabling flexibility in evolving deliverables. Develop an award fee structure and procedures based on objective criteria for evaluating subcontractors' performance in:**
 - **Schedule Preservation**
 - **Cost Containment**
 - **Technical Performance**
 - **Architecture and COTS Compatibility**
 - **Continuous Integration Support**
 - **Program Management**
 - **Risk Management**



Architecture Challenges #5 & #6:

COTS Upgrade Synchronization and Obsolescence

- **Risk #5: Many subcontractors means a proliferation of evolving COTS interfaces**
- **Risk #6: Aggressively-bid subcontracts can lead to delivery of obsolete COTS**
 - **New COTS released every 8-9 months (GSAW)**
 - **COTS unsupported after 3 releases (GSAW)**
 - **An actual delivery: 120 COTS; 46% unsupported**
- **Strategy #5: Emphasize COTS interoperability in source selection process**
- **Strategy #6: Contract provisions ensuring delivery of refreshed COTS products.**



Conclusions: CSOS and Software Architectures

- **Software is a critical enabling technology for Complex Systems of Systems(CSOS)**
 - It provides the means for dealing with many CSOS risks and opportunities
 - Particularly those involving interoperability and rapid adaptability to change
- **Software's CSOS benefits come with their own risks**
 - Some of these are rarely encountered in hardware-intensive acquisitions
 - They particularly affect CSOS software architecting
- **Strategies for dealing with CSOS software risks are becoming available**
 - Some require architecture technology advances
 - Some require changes in traditional acquisition practices
 - Some require escalation of software architecture management authority