Current and Future Challenges for Software Cost Estimation and Data Collection

Barry Boehm, USC-CSSE

GSAW 2010 Cost Data Workshop

March 3, 2010

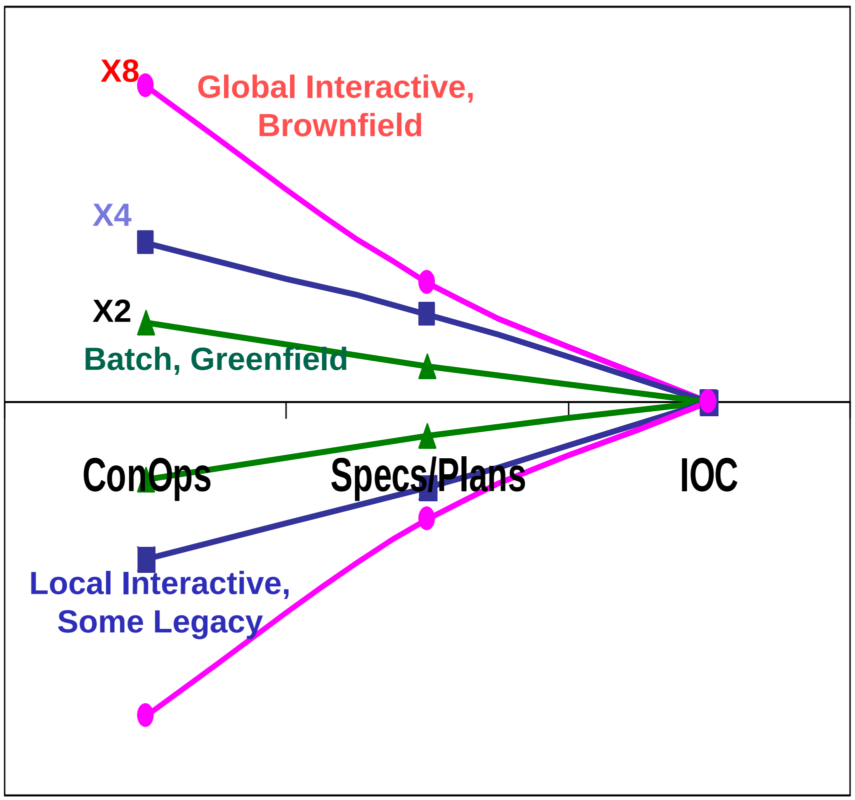
Summary

- **Current and future trends create challenges for DoD software data collection and analysis**
  - **Mission challenges: emergent requirements, rapid change, net-centric systems of systems, COTS and services, high assurance with agility**
  - **DoD initiatives: DoDI 5000.02, evolutionary acquisition, competitive prototyping, Software Resources Data Reports**
- **Updated software data definitions and estimation methods could help DoD systems management**
  - **Examples: incremental and evolutionary development; COTS and services; net-centric systems of systems**
  - **Further effort and coordination needed to converge on these**

Current and Future DoD Challenges

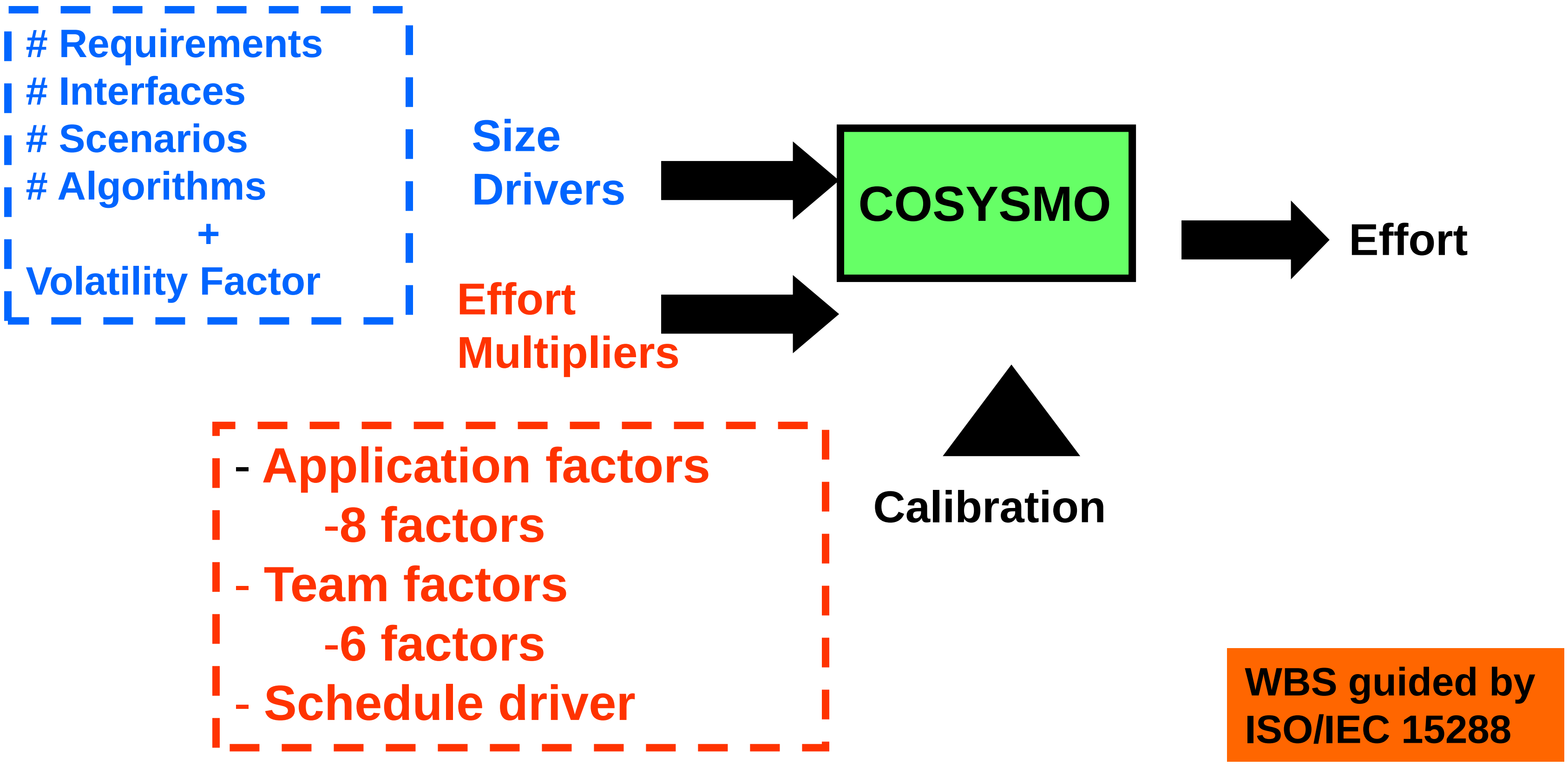
- Emergent requirements
  - Cannot prespecify requirements, cost, schedule, EVMS
  - Need to estimate and track early concurrent engineering
- Rapid change
  - Long acquisition cycles breed obsolescence
  - DoDI 5000.02 emphasis on evolutionary acquisition
- Net-centric systems of systems
  - Incomplete visibility and control of elements
- Model, COTS, service-based, Brownfield systems
  - New phenomenology, counting rules
- Always-on, never-fail systems
  - Need to balance agility and high assurance

The Broadening Early Cone of Uncertainty (CU)



- Need greater investments in narrowing CU
  - Mission, investment, legacy analysis
  - Competitive prototyping
  - Concurrent engineering
  - Associated estimation methods and management metrics
- Larger systems will often have subsystems with narrower CU's

COSYSMO Operational Concept



TOC	COSYSMO Application Factor Selection											See Embedded Comments for Descriptions and Selection Criteria
COSYSMO Application Factor Description	Identifier	Current Prod. Range	Suggested Prod. Range	VLOW (VL)	LOW (L)	NOM (N)	HIGH (H)	VHIGH (VH)	XHIGH (XH)	Rating Selected	Resulting Multiplier	Application Factor Rating Selection Comments
Requirements Understanding	RQMT	1.73	1.73	1.40	1.20	1.00	0.90	0.81	****	N	1.00	
Architecture Complexity	ARCH	1.66	1.66	1.28	1.14	1.00	0.88	0.77	****	N	1.00	
Level of Service (KPP) Requirements	LSVC	2.50	2.50	0.66	0.83	1.00	1.33	1.65	****	N	1.00	
Migration Complexity	MIGR	1.50	1.50	****	****	1.00	1.25	1.50	****	N	1.00	
No. and Diversity of Installations/Platforms	INST	1.50	1.50	****	****	1.00	1.25	1.50	****	N	1.00	
No. of Recursive Levels in the Design	RECU	1.50	1.50	0.82	0.91	1.00	1.12	1.23	****	N	1.00	
Documentation to Match Lifecycle Needs	DOCU	0.67	0.67	0.82	0.91	1.00	1.12	1.23	****	N	1.00	
Technology Maturity	TMAT	2.50	2.50	1.75	1.37	1.00	0.85	0.70	****	N	1.00	

Productivity Range (PR) is the Highest Number / Lowest Number and is an indication of the "Relative Degree of Influence" of this parameter on SE effort as currently

The "Suggested" column has no immediate impact in the COSYSMO SE Costing Mode. However, for the COSYSMO SE Data Collection Mode, it serves as a means of collecting your inputs as to what you think the "Relative Degree of Influence" of this parameter should be based upon your overall experience (not specific to the past program being characterized). If you agree with the "Current" number, do nothing. If you disagree, simply overwrite the current number with a new number n (n>1.0) in the appropriate cell.

Select the Rating from the pullc that best represents the Rating program being estimated in the Mode or in the SE Data Collectic Rating that best characterizes t program for which you are prov

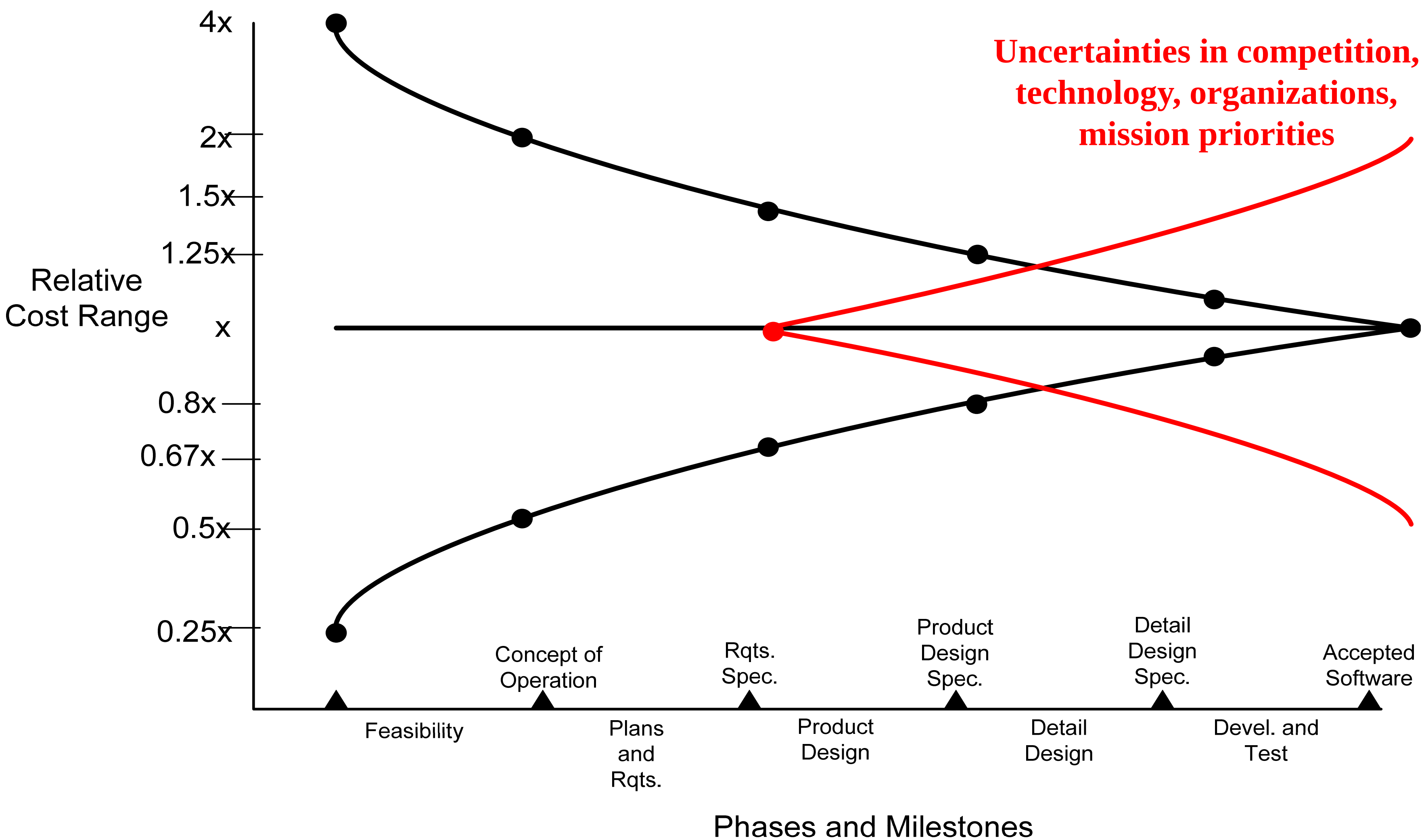
4. Parameters I | 5. Parameters II | 6a. Staffing Table | 6b. Staffing Chart | 7. Labor Distribution | Local SE Data Repository | **8a. Application Factors** | 8t

Next-Generation Systems Challenges

- **Emergent requirements**
  - Example: Virtual global collaboration support systems
  - Need to manage early concurrent engineering
- **Rapid change**
  - In competitive threats, technology, organizations, environment
- **Net-centric systems of systems**
  - Incomplete visibility and control of elements
- **Model, COTS, service-based, Brownfield systems**
  - New phenomenology, counting rules
- **Always-on, never-fail systems**
  - Need to balance agility and high assurance

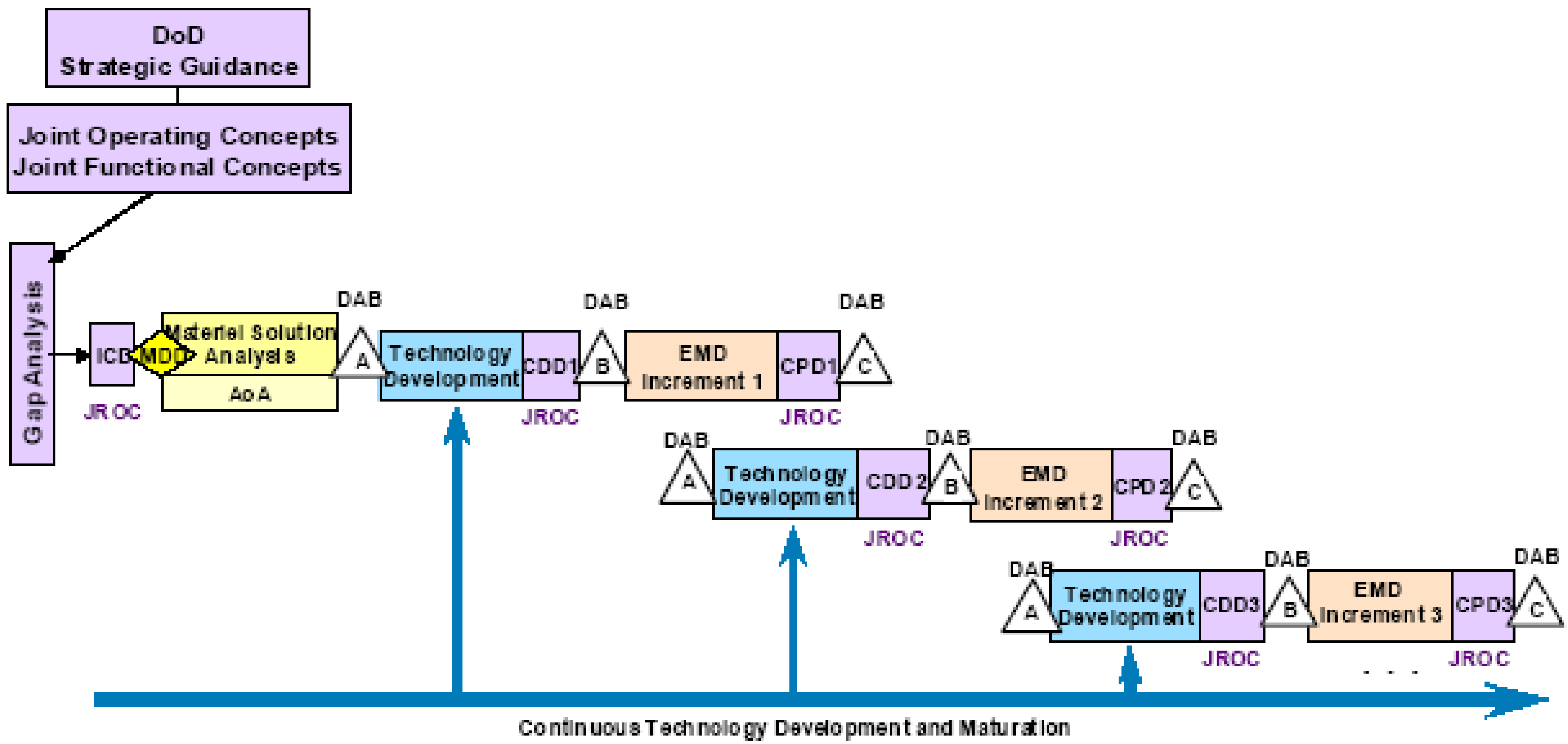
Rapid Change Creates a Late Cone of Uncertainty

– Need evolutionary/incremental vs. one-shot development



Evolutionary Acquisition per New DoDI 5000.02

No clean boundary between R&D and O&M



Incremental Development Productivity Decline (IDPD)

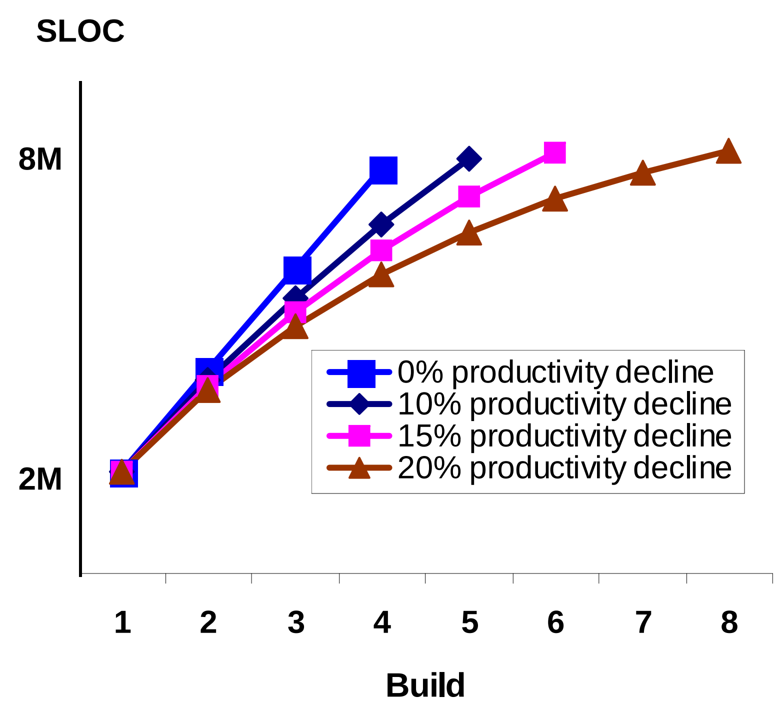
- **Example: Site Defense BMD Software**
  - 5 builds, 7 years, $100M; operational and support software
  - Build 1 productivity over 300 LOC/person month
  - Build 5 productivity under 150 LOC/PM
    - Including Build 1-4 breakage, integration, rework
    - 318% change in requirements across all builds
    - IDPD factor = 20% productivity decrease per build
  - Similar trends in later unprecedented systems
  - Not unique to DoD: key source of Windows Vista delays
- **Maintenance of full non-COTS SLOC, not ESLOC**
  - Build 1: 200 KSLOC new; 200K reused@20% = 240K ESLOC
  - Build 2: 400 KSLOC of Build 1 software to maintain, integrate

IDPD Cost Drivers: Conservative 4-Increment Example

- **Some savings: more experienced personnel (5-20%)**
  - **Depending on personnel turnover rates**
- **Some increases: code base growth, diseconomies of scale, requirements volatility, user requests**
  - **Breakage, maintenance of full code base (20-40%)**
  - **Diseconomies of scale in development, integration (10-25%)**
  - **Requirements volatility; user requests (10-25%)**
- **Best case: 20% more effort (IDPD=6%)**
- **Worst case: 85% (IDPD=23%)**

Effects of IDPD on Number of Increments

- **Model relating productivity decline to number of builds needed to reach 8M SLOC Full Operational Capability**
- **Assumes Build 1 production of 2M SLOC @ 100 SLOC/PM**
  - **20000 PM/ 24 mo. = 833 developers**
  - **Constant staff size for all builds**
- **Analysis varies the productivity decline per build**
  - **Extremely important to determine the incremental development productivity decline (IDPD) factor per build**



Incremental Development Data Challenges

- Breakage effects on previous increments
  - Modified, added, deleted SLOC: need Code Count with diff tool
- Accounting for breakage effort
  - Charged to current increment or I&T budget (IDPD)
    - IDPD effects may differ by type of software
  - "Breakage ESLOC" added to next increment
  - Hard to track phase and activity distributions
    - Hard to spread initial requirements and architecture effort
- Size and effort reporting
  - Often reported cumulatively
  - Subtracting previous increment size may miss deleted code
- Time-certain development
  - Which features completed? (Fully? Partly? Deferred?)

"Equivalent SLOC" Paradoxes

- Not a measure of software size
- Not a measure of software effort
- Not a measure of delivered software capability
- A quantity derived from software component sizes and reuse factors that helps estimate effort
- Once a product or increment is developed, its ESLOC loses its identity
  - Its size expands into full SLOC
  - Can apply reuse factors to this to determine an ESLOC quantity for the next increment
    - But this has no relation to the product's size

Current and Future DoD Challenges

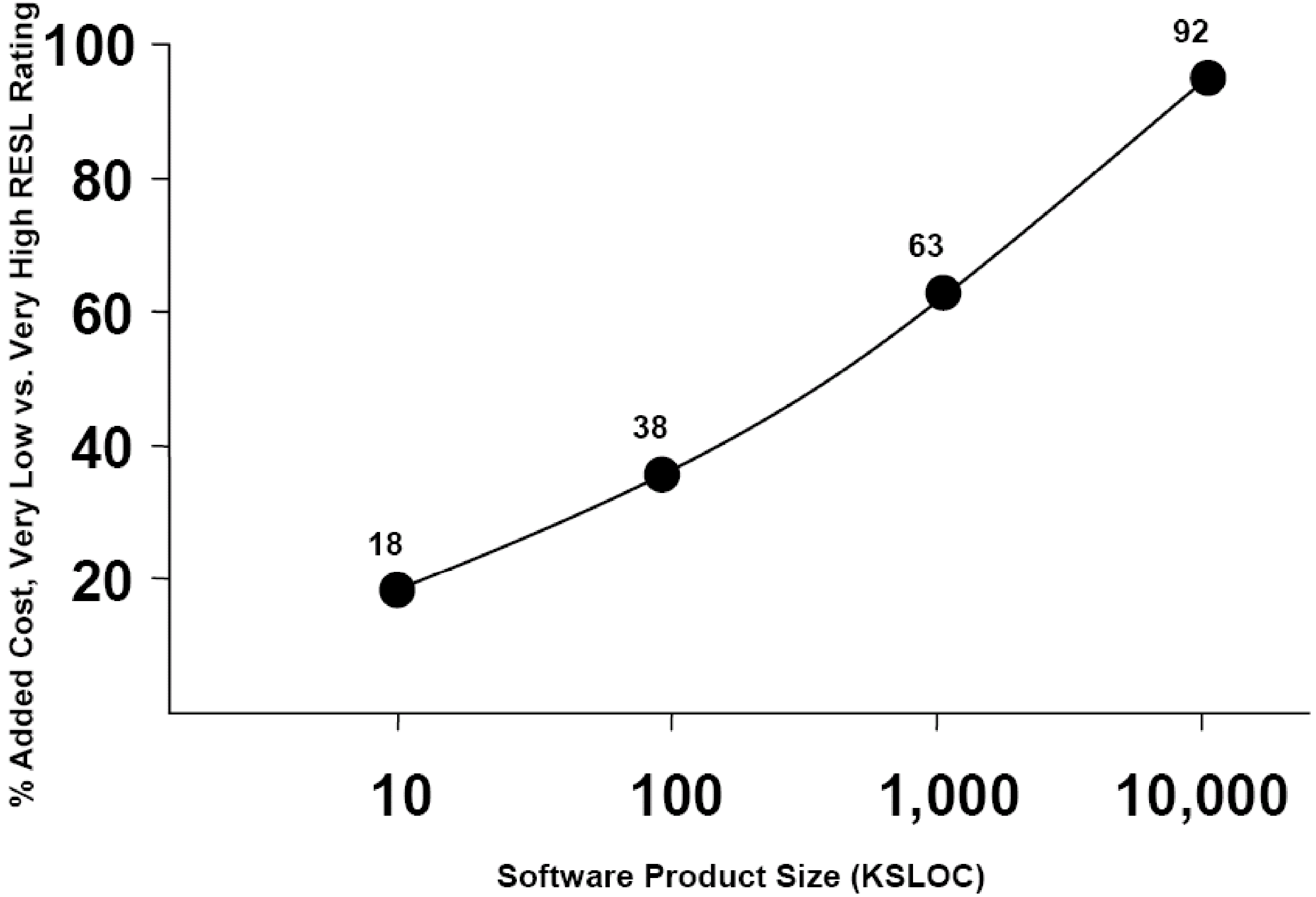
- Emergent requirements
  - Cannot prespecify requirements, cost, schedule, EVMS
  - Need to estimate and track early concurrent engineering
- Rapid change
  - Long acquisition cycles breed obsolescence
  - DoDI 5000.02 emphasis on evolutionary acquisition
- Net-centric systems of systems
  - Incomplete visibility and control of elements
- Model, COTS, service-based, Brownfield systems
  - New phenomenology, counting rules
- Always-on, never-fail systems
  - Need to balance agility and high assurance

Net-Centric Systems of Systems Challenges

- Need for rapid adaptation to change
  - See first, understand first, act first, finish decisively
- Built-in authority-responsibility mismatches
  - Increasing as authority decreases through Directed, Acknowledged, Collaborative, and Virtual SoS classes
- Severe diseconomies of scale
  - Weak early architecture and risk resolution
  - Need thorough flowdown/up of estimates, actuals
  - More complex integration and test preparation, execution
- More software intensive
  - Best to use parallel software WBS
- Many different classes of system elements
  - One-size-fits-all cost models a poor fit

Added Cost of Weak Architecting

Calibration of COCOMO II Architecture and Risk Resolution factor to 161 project data points



Model, COTS, Service-Based, Brownfield Systems

New phenomenology, counting rules

- **Product generation from model directives**
  - Treat as very high level language: count directives
- **Sizing COTS and services use needs improvement**
  - Unrealistic to use COTS, services SLOC for sizing
  - Alternatives: function point elements, amount of glue code, activity-based assessment costing, tailoring parameters
- **Brownfield legacy constraints, re-engineering**
  - Re-engineer legacy code to fit new architecture
  - Apply reuse model for re-engineering
- **A common framework for reuse, incremental development, maintenance, legacy re-engineering?**
  - All involve reusing, modifying, deleting existing software

Data definition topics for discussion

- Ways to treat data elements
  - COTS, other OTS (open source; services; GOTS; reuse; legacy code)
  - Other size units (function points object points, use case points, etc.)
  - Generated code: counting generator directives
  - Requirements volatility
  - Rolling up CSCIs into systems
  - Cost model inputs and outputs (e.g., submitting estimate files)
- Scope issues
  - Cost drivers, Scale factors
  - Reuse parameters: Software Understanding , Programmer Unfamiliarity
  - Phases included: hardware-software integration; systems of systems integration, transition, maintenance
  - WBS elements and labor categories included
  - Parallel software WBS
- How to involve various stakeholders
  - Government, industry, commercial cost estimation organizations

Summary

- **Current and future trends create challenges for DoD software data collection and analysis**
  - **Mission challenges: emergent requirements, rapid change, net-centric systems of systems, COTS and services, high assurance with agility**
  - **DoD initiatives: DoDI 5000.02, evolutionary acquisition, competitive prototyping, Software Resources Data Reports**
- **Updated software data definitions and estimation methods could help DoD systems management**
  - **Examples: incremental and evolutionary development; COTS and services; net-centric systems of systems**
  - **Further effort and coordination needed to converge on these**

References

Boehm, B., "Some Future Trends and Implications for Systems and Software Engineering Processes", *Systems Engineering* 9(1), pp. 1-19, 2006.

Boehm, B. and Lane J., "21st Century Processes for Acquiring 21st Century Software-Intensive Systems of Systems." *CrossTalk*: Vol. 19, No. 5, pp.4-9, 2006.

Boehm, B., and Lane, J., "Using the ICM to Integrate System Acquisition, Systems Engineering, and Software Engineering," *CrossTalk*, October 2007, pp. 4-9.

Boehm, B., Brown, A.W.. Clark, B., Madachy, R., Reifer, D., et al., *Software Cost Estimation with COCOMO II*, Prentice Hall, 2000.

Dahmann, J. (2007); "Systems of Systems Challenges for Systems Engineering", Systems and Software Technology Conference, June 2007.

Department of Defense (DoD), *Defense Acquisition Guidebook, version 1.6, http://akss.dau.mil/dag/, 2006*.

Department of Defense (DoD), *Instruction 5000.2, Operation of the Defense Acquisition System,* May 2003.

Department of Defense (DoD), *Systems Engineering Plan Preparation Guide, USD(AT&L),* 2004.

Galorath, D., and Evans, M., Software Sizing, Estimation, and Risk Management, Auerbach, 2006.

Lane, J. and Boehm, B., "Modern Tools to Support DoD Software-Intensive System of Systems Cost Estimation, DACS State of the Art Report, also Tech Report USC-CSSE-2007-716

Lane, J., Valerdi, R., "Synthesizing System-of-Systems Concepts for Use in Cost Modeling," *Systems Engineering*, Vol. 10, No. 4, December 2007.

Madachy, R., "Cost Model Comparison," Proceedings 21st, COCOMO/SCM Forum, November, 2006, http://csse.usc.edu/events/2006/CIIForum/pages/program.html

Maier, M., "Architecting Principles for Systems-of-Systems"; *Systems Engineering*, Vol. 1, No. 4 (pp 267-284).

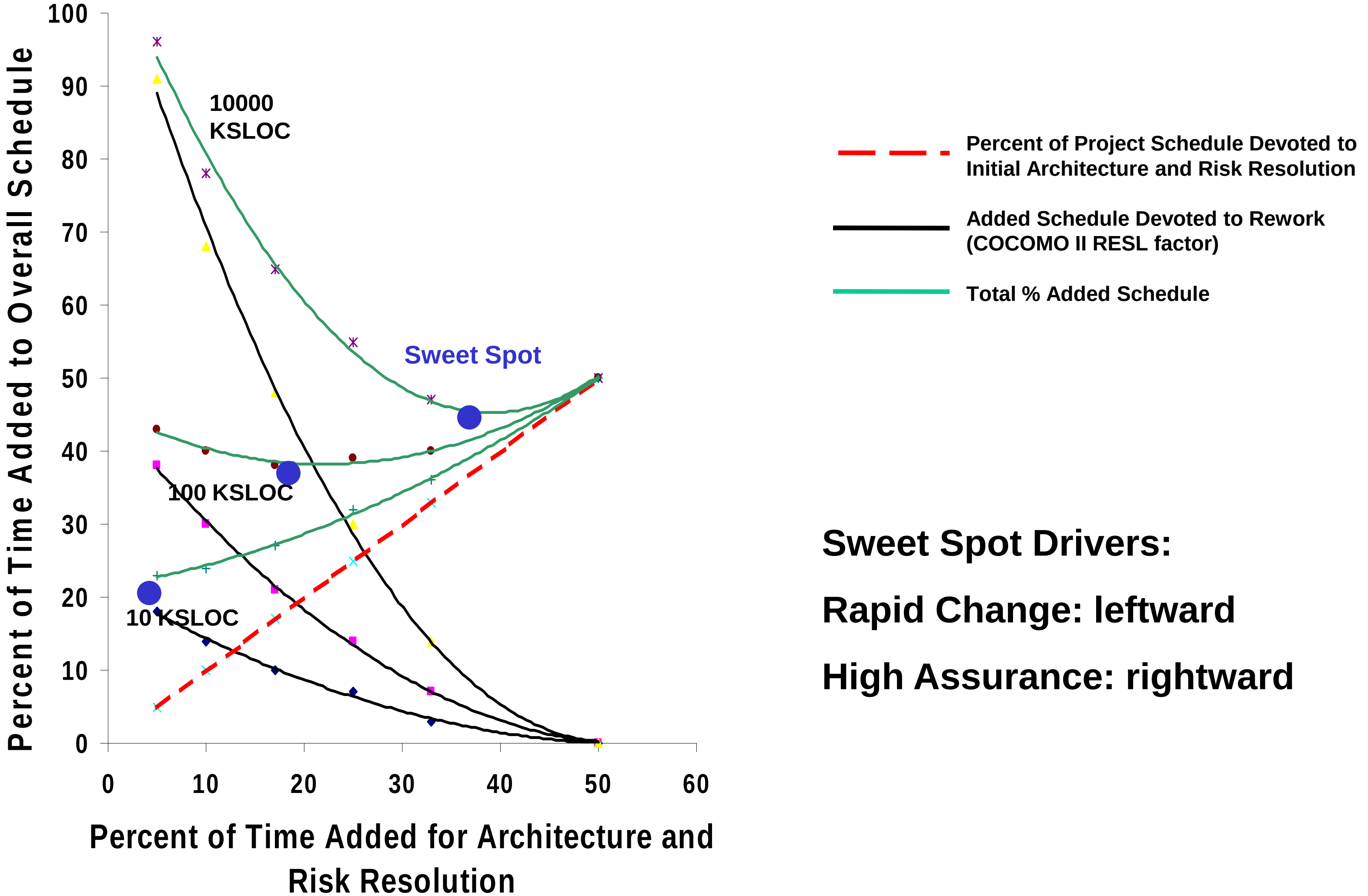
Northrop, L., et al., *Ultra-Large-Scale Systems: The Software Challenge of the Future*, Software Engineering Institute, 2006.

Reifer, D., "Let the Numbers Do the Talking," *CrossTalk*, March 2002, pp. 4-8.

Valerdi, R, *Systems Engineering Cost Estimation with COSYSMO*, Wiley, 2009 (to appear)

Backup Charts

How Much Architecting is Enough?
- Larger projects need more

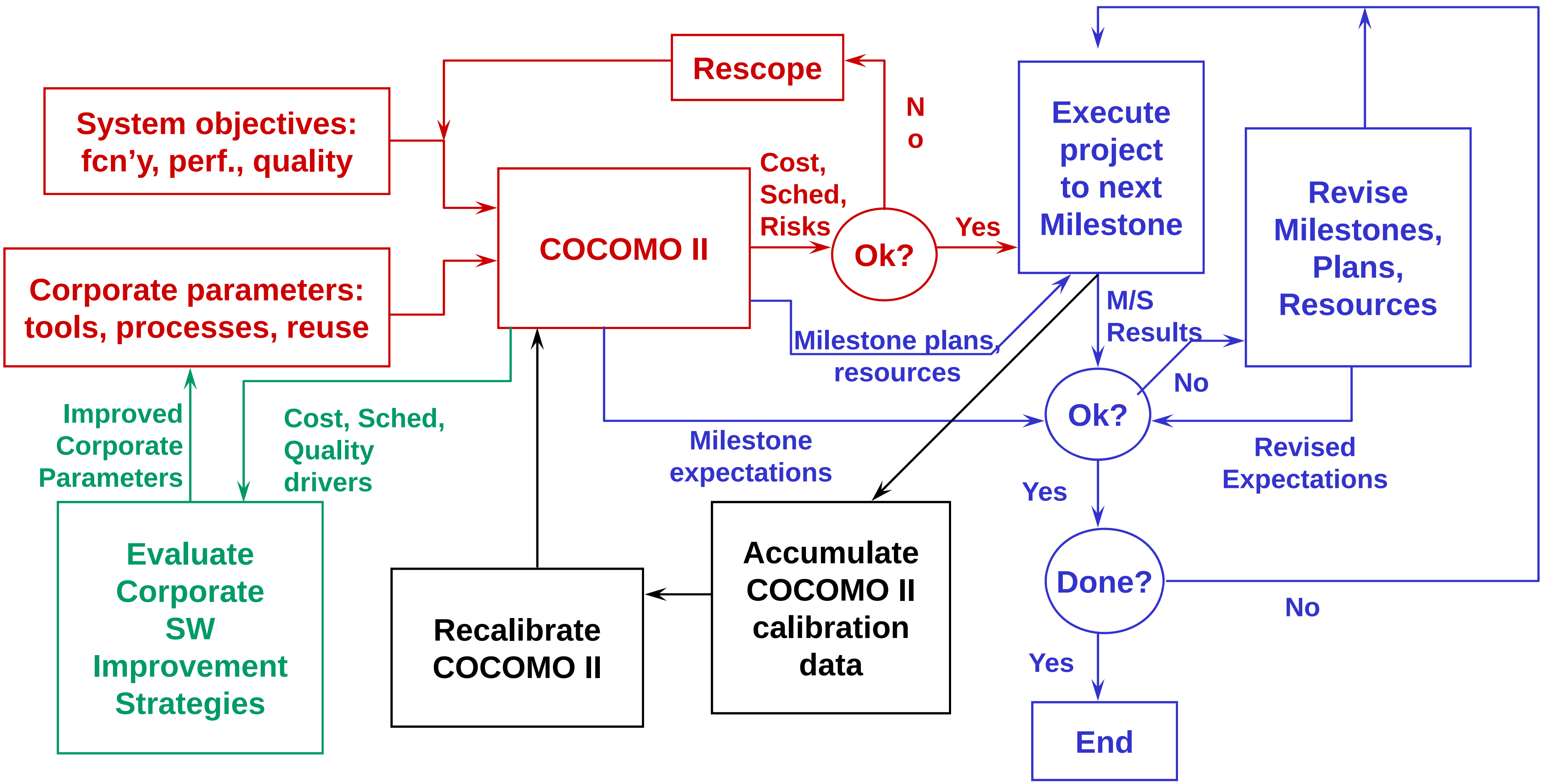


Sweet Spot Drivers:

Rapid Change: leftward

High Assurance: rightward

TRW/COCOMO II Experience Factory: IV



Choosing and Costing Incremental Development Forms

Type	Examples	Pros	Cons	Cost Estimation
Evolutionary Sequential	Small: Agile Large: Evolutionary Development	Adaptability to change	Easiest-first; late, costly breakage	Small: Planning-poker-type Large: Parametric with IDPD
Prespecified Sequential	Platform base plus PPPIs	Prespecifiable full-capability requirements	Emergent requirements or rapid change	COINCOMO with no increment overlap
Overlapped Evolutionary	Product lines with ultrafast change	Modular product line	Cross-increment breakage	Parametric with IDPD and Requirements Volatility
Rebaselining Evolutionary	Mainstream product lines; Systems of systems	High assurance with rapid change	Highly coupled systems with very rapid change	COINCOMO, IDPD for development; COSYSMO for rebaselining

IDPD: Incremental Development Productivity Decline, due to earlier increments breakage, increasing code base to integrate

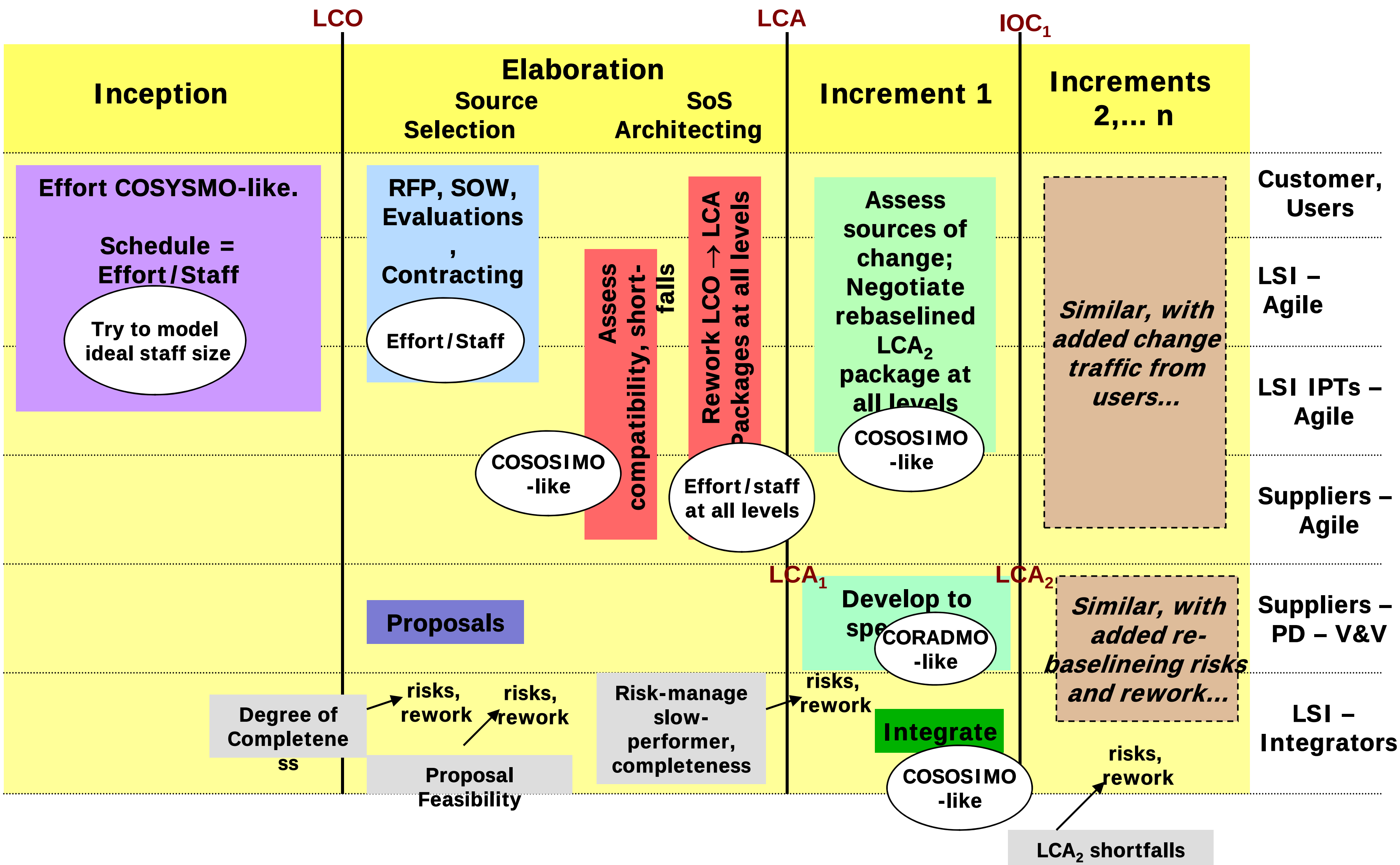
PPPIs: Pre-Planned Product Improvements

COINCOMO: COCOMO Incremental Development Model (COCOMO II book, Appendix B)

COSYSMO: Systems Engineering Cost Model (in-process COSYSMO book)

All Cost Estimation approaches also include expert-judgment cross-check.

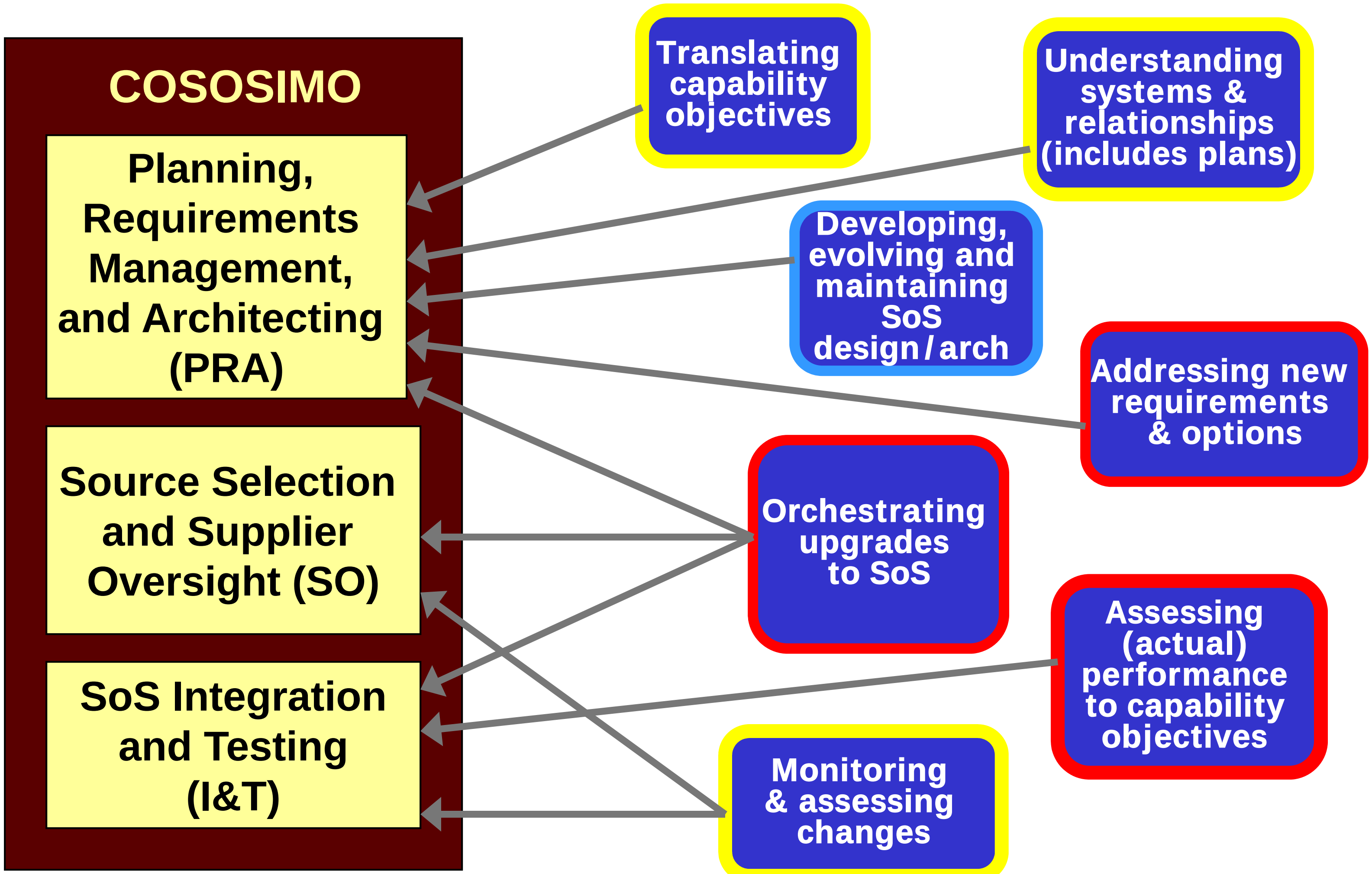
Compositional approaches: Directed systems of systems



Comparison of Cost Model Parameters

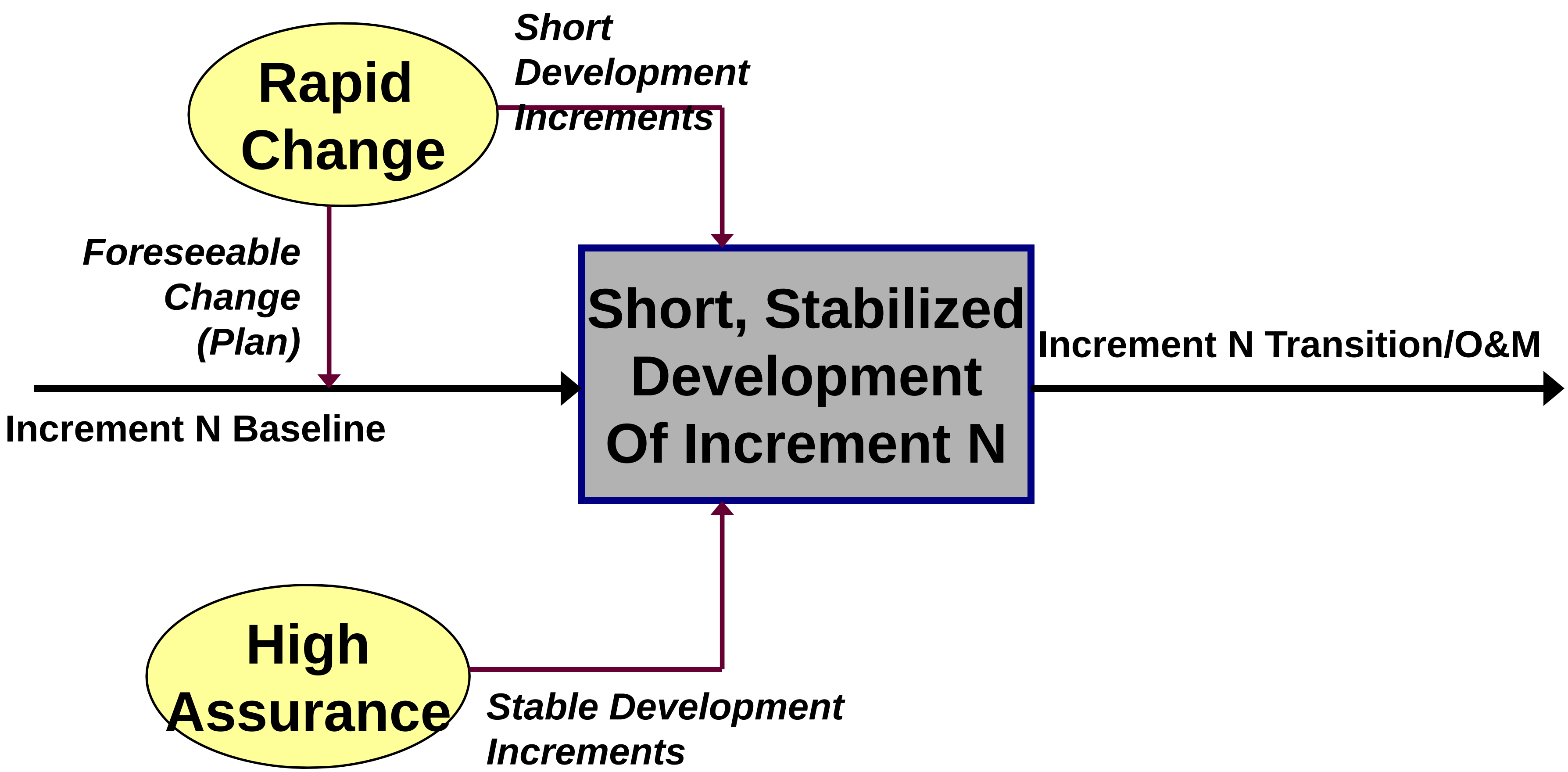
Parameter Aspects	COSYSMO	COSOSIMO
Size drivers	# of system requirements # of system interfaces # operational scenarios *# algorithms*	# of SoS requirements # of SoS interface protocols *# of constituent systems* *# of constituent system organizations* # operational scenarios
"Product" characteristics	Size/complexity Requirements understanding Architecture understanding Level of service requirements *# of recursive levels in design* *Migration complexity* Technology risk *#/ diversity of platforms/installations* *Level of documentation*	Size/complexity Requirements understanding Architecture understanding Level of service requirements *Component system maturity and stability* *Component system readiness*
Process characteristics	Process capability *Multi-site coordination* Tool support	Maturity of processes Tool support *Cost/schedule compatibility* SoS risk resolution
People characteristics	Stakeholder team cohesion Personnel/team capability Personnel experience/continuity	Stakeholder team cohesion SoS team capability

SoSE Core Element Mapping to COSOSIMO Sub-models



Achieving Agility and High Assurance -I

Using timeboxed or time-certain development
Precise costing unnecessary; feature set as dependent variable



Achieving Agility and High Assurance -II