Estimating System-of-System Architecture Definition and Integration Effort

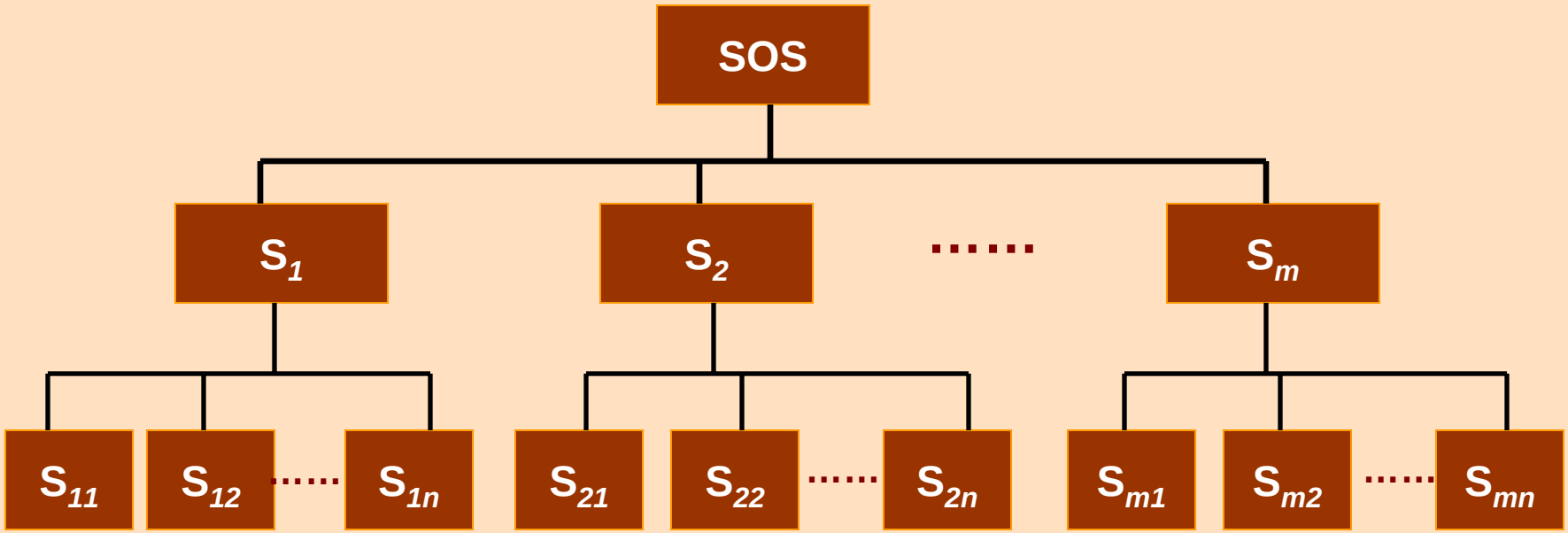
Jo Ann Lane
University of Southern California
Center for Software Engineering

PSM July 2004

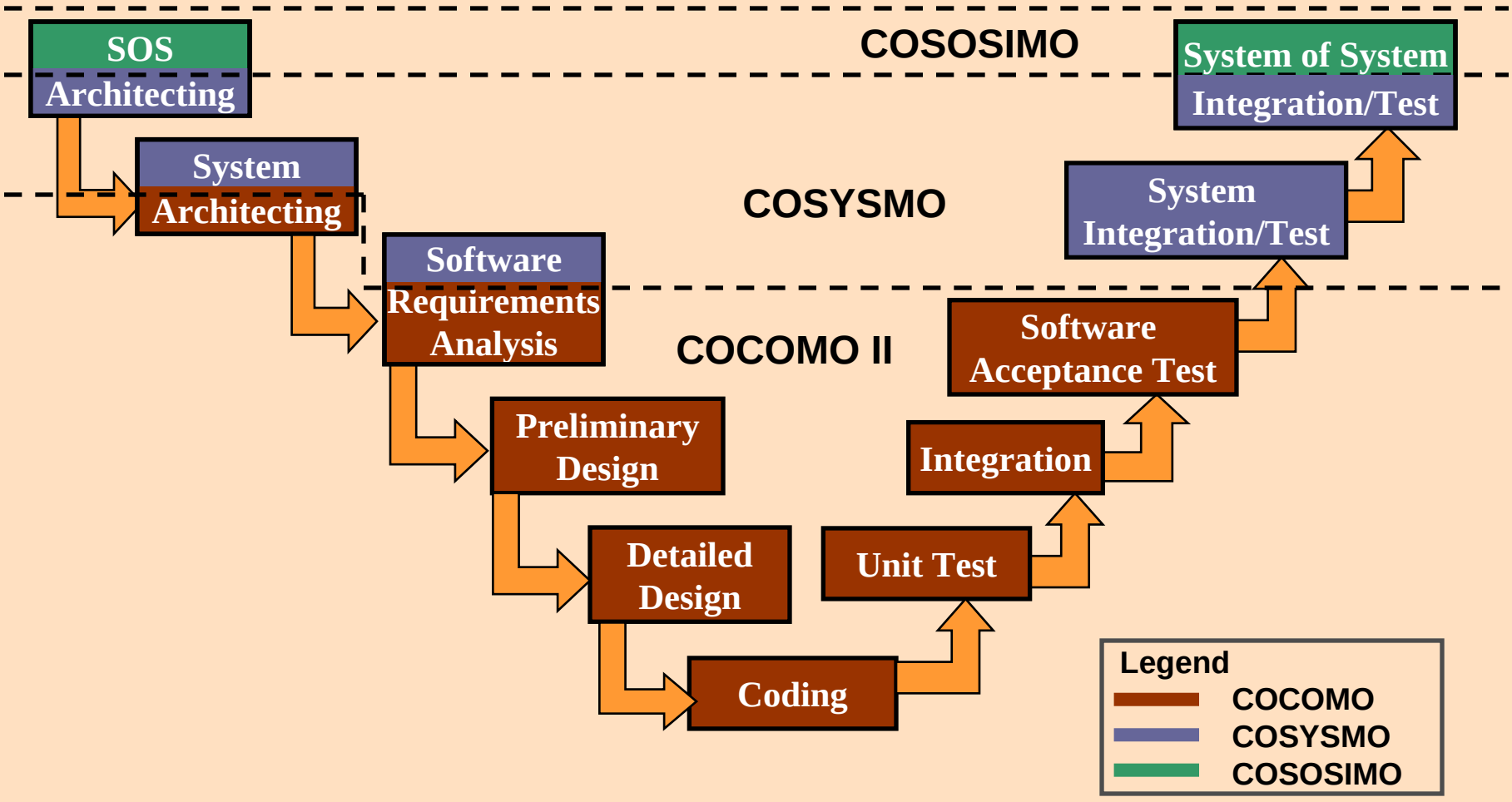
Goals of This Presentation

- Provide an overview of
  - System-of-system concepts
  - The desired system-of-system activities to be covered by the cost model
  - The cost model approaches, concepts, and definitions
  - Current issues/questions under investigation
- Present an example using current investigational version of cost model
- Solicit data for further model investigations and calibration
- Obtain feedback/suggestions on approach and data collection

System of Systems (SoS) Concept



High Level Partitioning of Cost Models



Constructive System-of-System Integration Cost Model (COSOSIMO)

- Parametric model to estimate the effort associated with the definition and integration of software-intensive “system of systems” components
- Includes at least one size driver and 6 exponential scale factors related to effort
- Targets input parameters that can be determined in early phases
- Goal is to have zero overlap with COCOMO II and COSYSMO

Key Activities Covered by Each Cost Model

COSOSIMO

- SoS architecture definition
- SoS integration activities
  - Development of SoS integration lab
  - Development of SoS level test plans and procedures
  - Execution of test SoS test procedures
  - High-level isolation of problems detected during integration

COSYSMO

- System/sub-system definition
  - Operational concepts
  - Operational scenarios
- System/sub-system elaboration
- System integration and test
- Resolution of system-level errors detected during test activities
- Deployment
- Maintenance

COCOMO II

- Application and system software development
  - Elaboration
  - Construction
- Development of test tools and simulators (estimated as a separate set of software)
- Resolution of software errors detected during test activities

Model Differences

COSOSIMO

- System of Systems architecture definition and integration
- Pre and Post COCOMO II effort
- Very new
- Only expert validation
- 6 exponential scale factors
- Candidate drivers
  - Effective KSLOC (eKSLOC)
  - Logical interfaces at SoS level

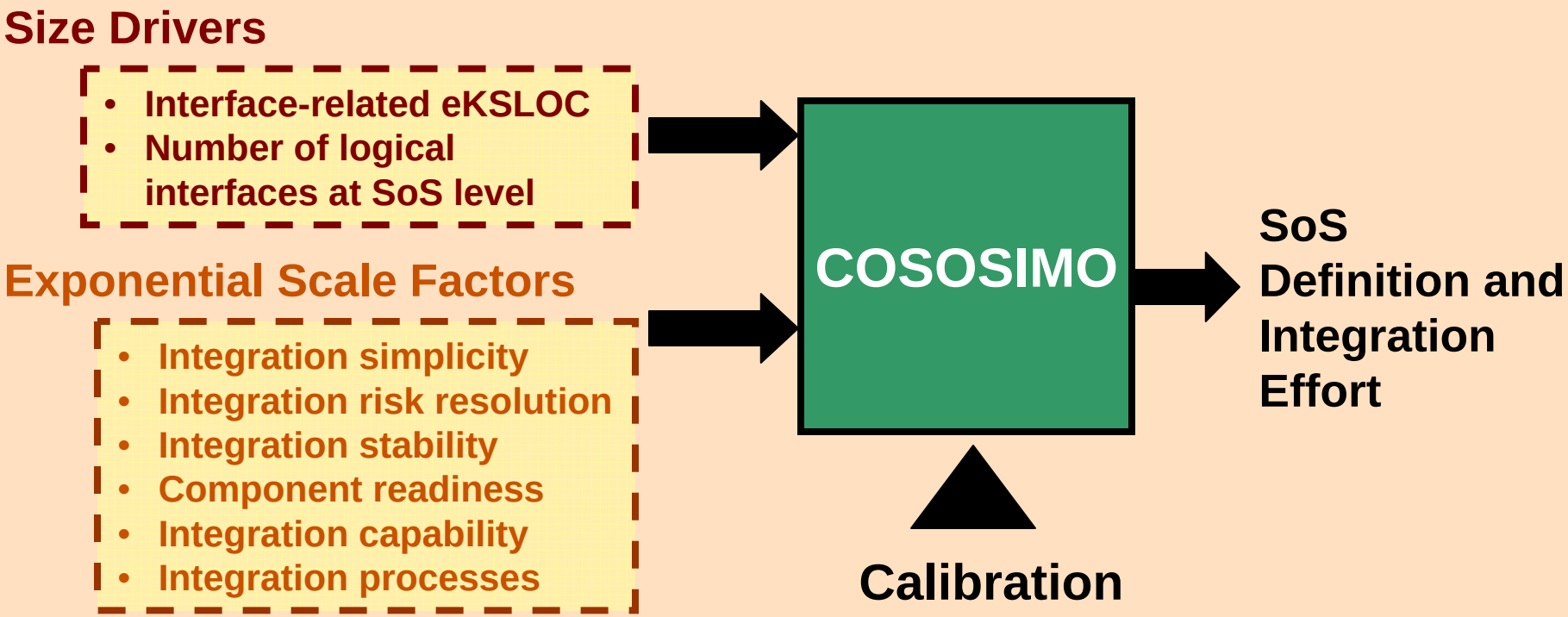
COSYSMO

- Systems engineering
- Entire life cycle
- 3 years old
- 11 calibration points
- 18 drivers
- Size is driven by
  - requirements
  - interfaces
  - algorithms
  - operational scenarios

COCOMO II

- Software development
- Development phases
- 20+ years old
- 161 calibration points
- 23 drivers
- Size is driven by effective SLOC (eSLOC)

COSOSIMO Operational Concept

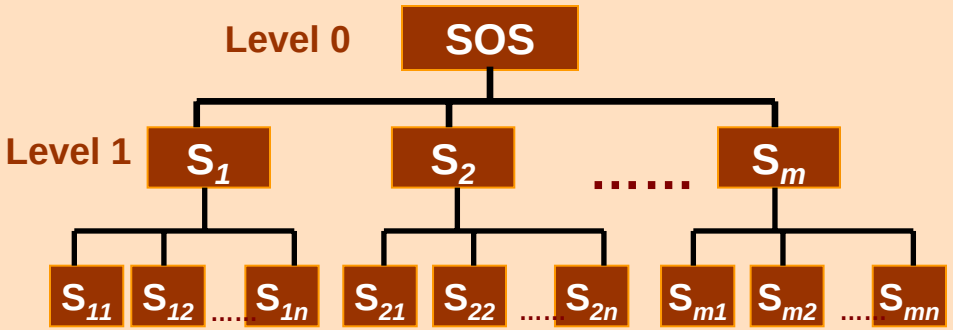


COSOSIMO Model Equations

$$\textit{Level 1}\ \text{IPM}\ (S_i) = A_i \left[\sum_{j=1}^{n_i} \text{Size}\ (S_{ij}) \right]^{B_i}$$

$$\textit{Level 0}\ \text{IPM}\ (\textit{SoS}) = A_0 \left[\sum_{i=1}^{m_i} \text{IPM}\ (S_i) \right]^{B_0}$$

Two level model that
- *First determines integration effort for first level subsystems….*
- *Then, using subsystem integration effort and SoS characteristics, determines SoS integration effort…*

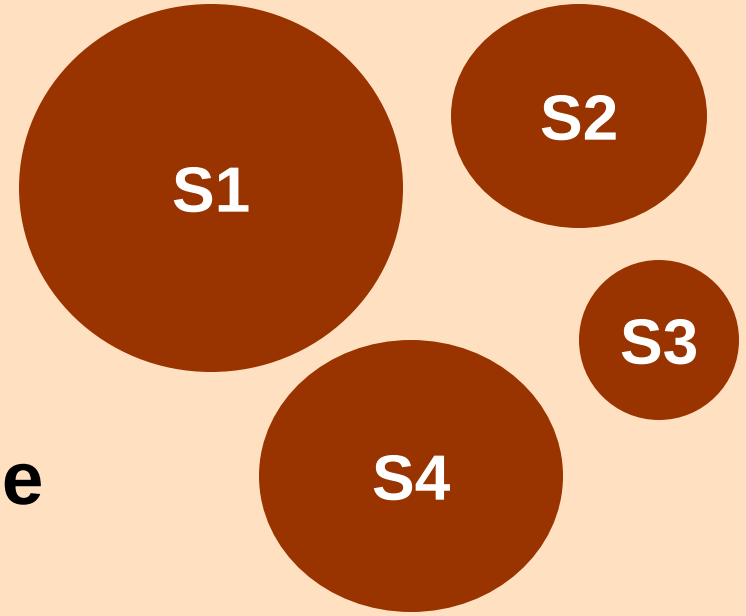


COSOSIMO Model Parameters

IPM	Integration effort in Person Months
S_i	The i^{th} subsystem within the SoS
A	Constant derived from historical project data
Size	Determined by computing the weighted average of the size driver(s)
n_i	Number of Subsystem level 2 components comprising the i^{th} subsystem
m	Number of Subsystem level 1 components comprising the SoS
B_i	Effort exponent for the i_{th} subsystem based on the subsystem's 6 exponential scale factors. The sum of the scale factors results in an overall exponential effort adjustment factor to the nominal effort.
B_0	Effort exponent for the *SoS* based on the SOS' 6 exponential scale factors. The sum of the scale factors results in an overall exponential effort adjustment factor to the nominal effort.

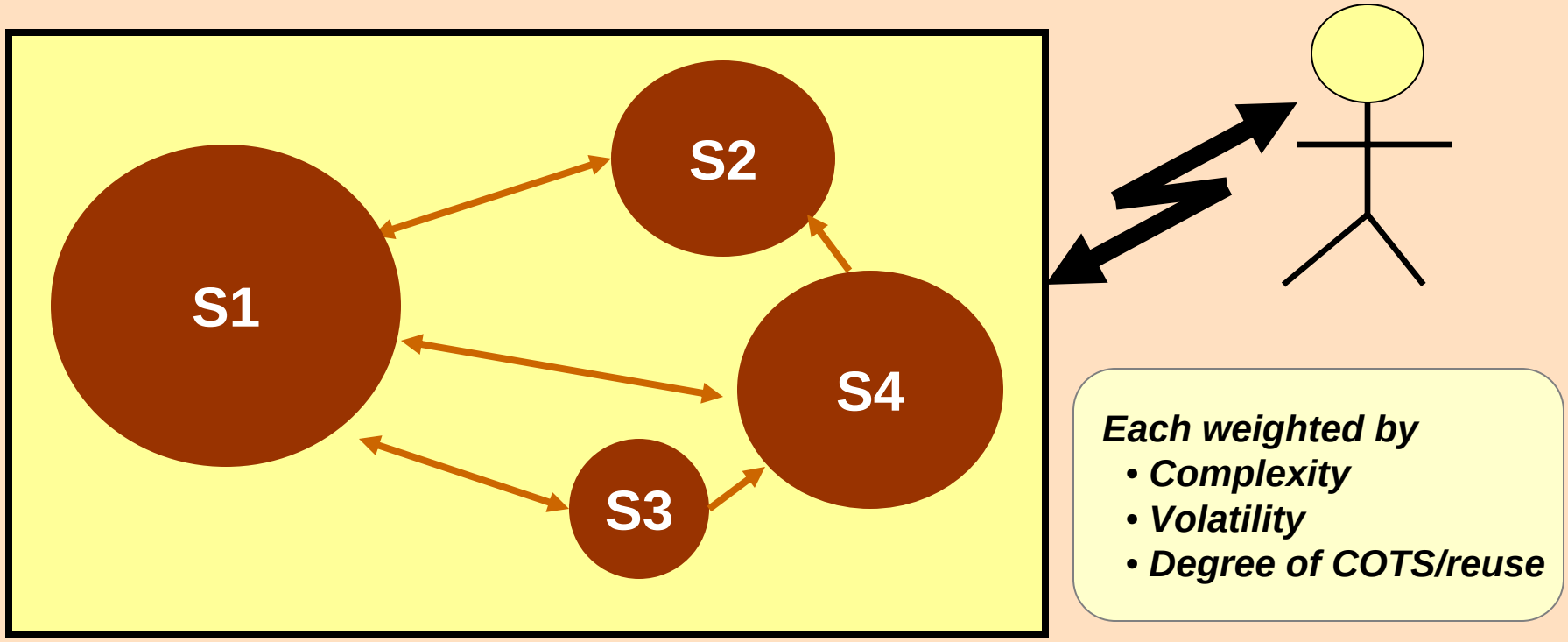
Current Level 1 Size Driver

- Subsystem development size measured in effective KSLOC (eKSLOC)
- eKSLOC can be calculated using COCOMO II
- Size weighted by
  - Complexity
  - Volatility
  - Degree of COTS/reuse



Additional Proposed Size Drivers

- Number of major interfaces
- Number of operational scenarios



Each weighted by
- *Complexity*
- *Volatility*
- *Degree of COTS/reuse*

Proposed Size Driver Definitions

Subsystem Software Size

This driver represents the software subsystem size. It is measured in terms of effective thousand lines of code (eKSLOC). eKSLOC can calculated using COCOMO II or a comparable estimation model or technique.

Easy	Nominal	Difficult
- Simple algorithms	- Straightforward, but non-trivial algorithms	- Complex algorithms
- Basic math	- Algebraic by nature	- Difficult math (calculus)
- Straightforward structure	- Nested structure with decision logic	- Recursive in structure with distributed control
- Simple data	- Relational data	- Persistent data
- Timing not an issue	- Timing a constraint	- Dynamic, with timing issues

Number of Major Interfaces

This driver represents the number of shared major physical and logical boundaries between subsystem components or functions (internal interfaces) and those external to the subsystem (external interfaces). These interfaces typically can be quantified by counting the number of interfaces identified in either the subsystem's context diagram and/or by counting the significant interfaces in all applicable Interface Control Documents.

Easy	Nominal	Difficult
- Well defined	- Loosely defined	- Ill defined
- Uncoupled	- Loosely coupled	- Highly coupled
- Cohesive	- Moderate cohesion	- Low cohesion
- Well behaved	- Predictable behavior	- Poorly behaved

Number of Operational Scenarios

This driver represents the number of operational scenarios that a system must satisfy. Such threads typically result in end-to-end test scenarios that are developed to validate the system and satisfy all of its requirements. The number of scenarios can typically be quantified by counting the number of unique end-to-end tests used to validate the system functionality and performance or by counting the number of high-level use cases developed as part of the operational architecture.

Easy	Nominal	Difficult
- Well defined	- Loosely defined	- Ill defined
- Loosely coupled	- Moderately coupled	- Tightly coupled or many dependencies/conflicting requirements
- Timelines not an issue	- Timelines a constraint	- Tight timelines through scenario network

Proposed Exponential Scale Factor Definitions

Integration Simplicity (ISMPL)

Represents a parameter which includes system component coupling, processing criticality, scope of key performance parameters, and system precedentedness.

Very Low	Low	Nominal	High	Very High	Extra High
▪ Very strong coupling ▪ Very strong criticality ▪ Cross-cutting key performance parameters ▪ Highly unprecedented	▪ Strong coupling ▪ Strong criticality ▪ Mostly unprecedented	▪ Both strong & weak coupling ▪ Mixed criticality ▪ Partly unprecedented	▪ Moderate coupling ▪ Moderate criticality ▪ Some new aspects	▪ Weak coupling ▪ Low criticality ▪ Few new aspects	▪ Very weak coupling ▪ No cross-cutting key performance parameters ▪ No new aspects

Integration Risk Resolution (IRESL)

Represents a multi-attribute parameter which includes number of integration risk items, risk management/mitigation plan, compatible schedules and budgets, expert availability, tool support, level of uncertainty in integration risk areas. IRESL is the subjective weighted average of the listed characteristics.

Characteristic	Very Low	Low	Nominal	High	Very High
▪ Number and criticality of risk items	▪ > 10 critical	▪ 5-10 critical	▪ 2-4 critical	▪ 1 critical	▪ <10 non-critical
▪ Risk mitigation activities	▪ None	▪ Little	▪ Some	▪ Risks generally covered	▪ Risks fully covered
▪ Schedule, budget, and internal milestones compatible with Risk Management Plan and integration scope	▪ None	▪ Little	▪ Some	▪ Generally	▪ Mostly
▪ % of top software system integrators available to project	▪ 20%	▪ 40%	▪ 60%	▪ 80%	▪ 100%
▪ Tool support available for tracking issues	▪ None	▪ Little	▪ Some	▪ Good	▪ Strong
▪ Level of uncertainty in integration risk area	▪ Extreme	▪ Significant	▪ Considerable	▪ Some	▪ Little

Integration Stability (ISBLY)

Indicates anticipated change in integration components during system of system integration activities.

Very Low	Low	Nominal	High	Very High	Extra High
▪ 10% change during integration period	▪ 7% change during integration period	▪ 4% change during integration period	▪ 2% change during integration period	▪ 1% change during integration period	▪ No change during integration period

Component Readiness (CREDY)

Indicates readiness of component (sub-component) for integration. Includes level of verification and validation (V&V) that has been performed prior to integration and level of subsystem integration activities that have been performed prior to integration into the SOSIL.

Very Low	Low	Nominal	High	Very High	Extra High
▪ Minimally V&V'd ▪ No pre-integration	▪ Some V&V ▪ Minimal pre-integration	▪ Moderate V&V ▪ Some pre-integration	▪ Considerable V&V ▪ Moderate pre-integration	▪ Extensive V&V ▪ Considerable pre-integration	▪ Thoroughly V&V'd ▪ Extensive pre-integration

Integration Capability (ICAPY)

Represents a multi-attribute parameter which includes the integration team cooperation and cohesion, integration personnel capability and continuity, and integration personnel experience (application, language, tool, and platform). ICAPY is the subjective weighted average of the listed characteristics.

Factor	Very Low	Low	Nominal	High	Very High	Extra High
▪ ITEAM	• Very difficult team interactions	• Some difficult team interactions	• Basically cooperative teams	• Largely cooperative teams	• Highly cooperative teams	▪ Seamless team interactions
▪ IPERS	• 35th percentile • 30% turnover rate	• 45th percentile • 20% turnover rate	• 55th percentile • 12% turnover rate	• 65th percentile • 9% turnover rate	• 75th percentile • 6% turnover rate	▪ 85th percentile • 4% turnover rate
▪ IPREX	• ≤ 5 months experience with app, lang, tools, platforms	• 9 months experience with app, lang, tools, platforms	• 1 year experience with app, lang, tools, platforms	• 2 years experience with app, lang, tools, platforms	• 4 years experience with app, lang, tools, platforms	• 6 years experience with app, lang, tools, platforms

Integration Processes (IPROC)

Represents a parameter that rates the maturity level and completeness of an integration team's integration processes, plans, and the SOS integration lab (SOSIL). IPROC is the subjective weighted average of the listed characteristics.

Very Low	Low	Nominal	High	Very High	Extra High
▪ Ad-hoc integration process ▪ Minimal SOSIL ▪ CMMI Level 1 (lower half)	▪ Minimal integration plans ▪ Immature core SOSIL ▪ CMMI Level 1 (upper half)	▪ Some integration plans ▪ Partly mature core SOSIL ▪ CMMI Level 2	▪ Moderate plans ▪ Mature core SOSIL ▪ CMMI Level 3	▪ Considerable plans ▪ Partly mature extended SOSIL ▪ CMMI Level 4	▪ Extensive plans ▪ Fully mature extended SOSIL ▪ CMMI Level 5

Current Issues

Issues and Questions Currently Under Investigation

- What is the best size driver
- If software size used
  - Should it be limited to the software performing interface operations
  - How should COTS product interfaces be accounted for
- If number of logical interfaces is used
  - Which ones to include
  - What level to count
  - How to specify complexities associated with various interfaces
- Do user scenarios and user interfaces capture additional size information needed to better estimate level of effort
- If multiple size drivers used, what is the relative weight of each
- Model outputs
  - Desired granularity of effort estimates
  - Associated schedule?

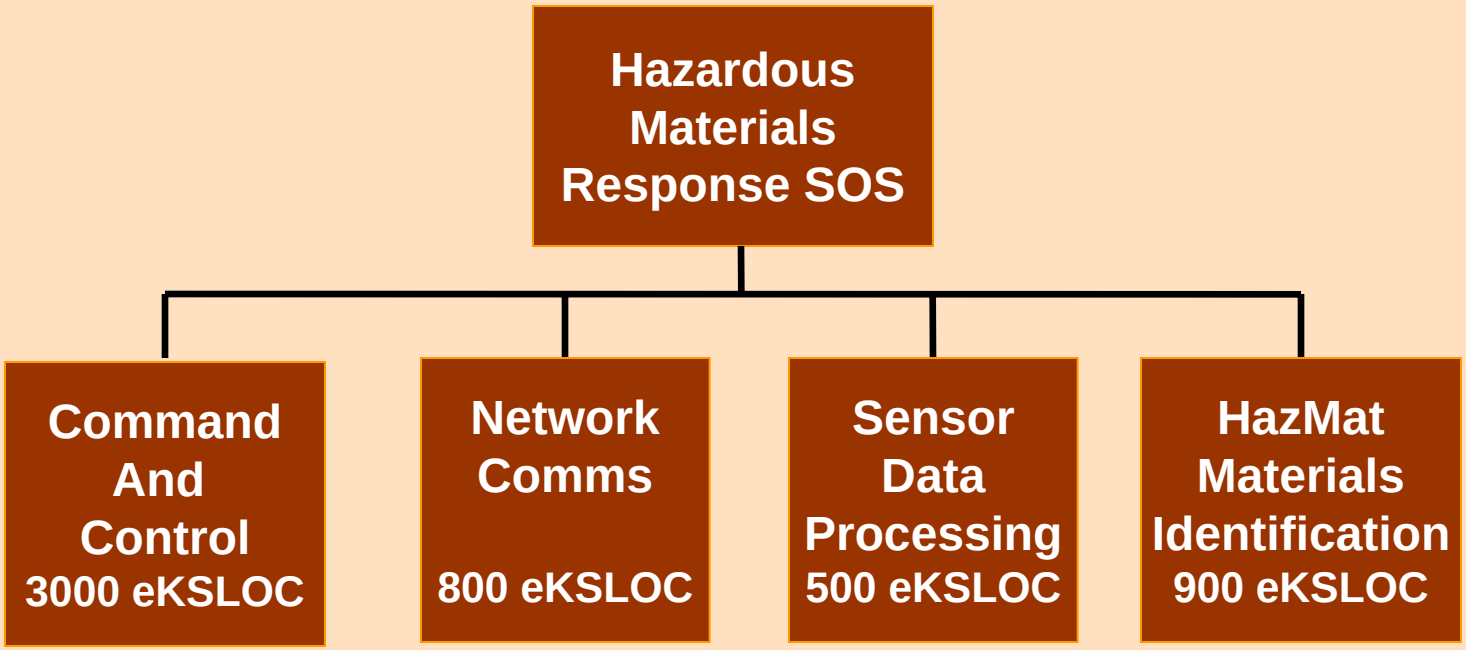
Issues and Questions Currently Under Investigation *(continued)*

- How to ensure no overlap with COSYSMO or COCOMO II models
- Are current scale factors
  - Relevant
  - Sufficient
- Are current scale factor values/range of values appropriate
- How well do the various model variations track with respect to
  - Expert judgment
  - Actual experiences/projects

SOS Estimation Example Using Only Software Size as the Size Driver

Hazardous Materials Response System of Systems

System of System Architecture Example



SOS Integration Calculations with Nominal Level 1 and Level 0 Drivers

Level 0	ISMPL	IRESL	ISBLY	CREDY	ICAPY	IPROC	B0
SOS	0.000	0.000	0.000	0.000	0.000	0.000	1.030

Level 1	ISMPL	IRESL	ISBLY	CREDY	ICAPY	IPROC	Bi
Command and Control	0.000	0.000	0.000	0.000	0.000	0.000	1.040
Network Comms	0.000	0.000	0.000	0.000	0.000	0.000	1.040
Sensor Data Processing	0.000	0.000	0.000	0.000	0.000	0.000	1.040
HazMat Material Identification	0.000	0.000	0.000	0.000	0.000	0.000	1.040

Level 1	eKSLOC	A1	Bi	IPMi
Command and Control	3000	1.000	1.040	4132.436
Network Comms	800	1.000	1.040	1045.234
Sensor Data Processing	600	1.000	1.040	774.956
HazMat Material Identification	900	1.000	1.040	1181.441

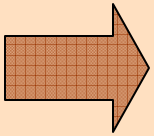
Level 0	Level 1 Effort Sum	A0	B0	IPM(SOS)
Sum	7134	1.000	1.030	9309.736

Total SOS Integration Effort: ~9310 Person Months or 775.8 Person Years

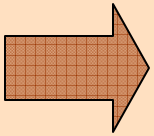
Potential Range of Values for Example

Case	Person Months	% of Total Estimated Nominal Development Effort
Nominal	9310	29%
Best	6477	20%
Worst	11907	37%

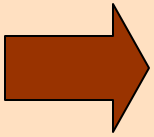
Parametric Cost Model Critical Path Tasks and Status



Converge on preliminary cost drivers, WBS



Converge on detailed definitions and rating scales



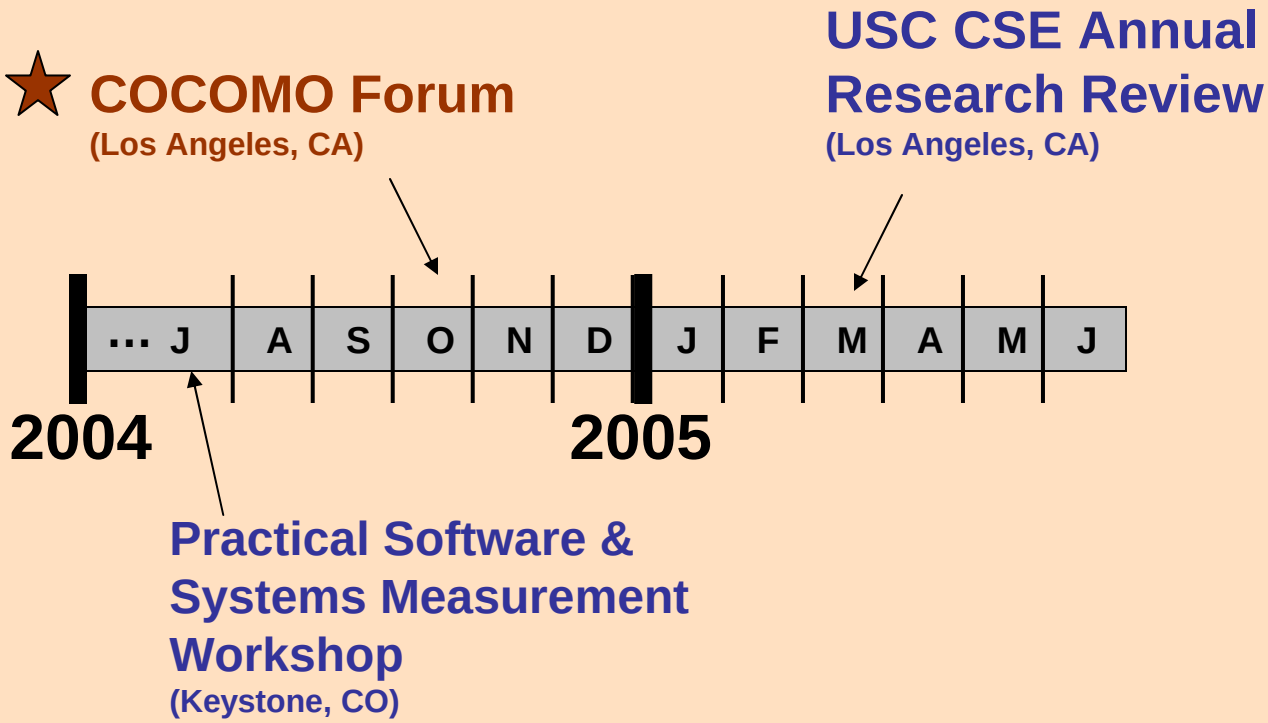
Obtain initial exploratory dataset

Refine model based on data collection and analysis experience

Obtain IOC calibration dataset

Refine IOC model and tool

Upcoming Calendar of Events: 2004/2005



Proposed First Working Group Meeting

Next Steps

- **Refine the model based on delphi inputs and actual data**
- **Working group meeting at October 2004 COCOMO II Workshop**

Questions or Comments?

- **Jo Ann Lane**

  jalane@tns.net
- **Websites**

  http://cse.usc.edu

  (COSOSIMO web site coming soon…)
- **Books**
  - **Boehm, B., et al, *Software Cost Estimation with COCOMOII*, 1st Ed, Prentice Hall, 2000**
  - **Boehm, B., *Software Engineering Economics*, 1st Ed, Prentice Hall, 1981**
- **Articles**
  - **Boehm, B., et al., *Future Trends, Implications in Cost Estimation Models*, CrossTalk April 2000.**
  - **Gilligan, John M., *Department of the Air Force, Military-Use Software: Challenges and Opportunities*, CrossTalk, January 2004.**