Many people have provided us with valuable insights on the challenge of integrating systems and software engineering, especially at the OSD/USC workshop in October 2007. We would particularly like to thank Bruce Amato (OSD), Elliot Axelband (Rand/USC), William Bail (Mitre), Kristen Baldwin (OSD), Kirstie Bellman (Aerospace), Winsor Brown (USC), Jim Cain (BAE Systems), David Castellano (OSD), George Friedman (USC), Brian Gallagher (CMU-SEI), Stuart Glickman (Lockheed Martin), Gary Hafen (Lockheed Martin), Dan Ingold (USC), Kelly Kim (Boeing), Sue Koolmanojwong (USC), DeWitt Latimer (USAF/USC), Rosalind Lewis (Aerospace), Azad Madni (ISTI), Mark Maier (Aerospace), Darrell Maxwell (USN), Ali Nikolai (SAIC), Karen Owens (Aerospace), Shawn Rahmani (Boeing), Bob Rassa (Raytheon), Don Reifer (RCI/USC), John Rieff (Raytheon), Wilson Rosa (USAF), Walker Royce (IBM), Kelly Schlegel (Boeing), Tom Schroeder (BAE Systems), David Seaver (Price Systems), Rick Selby (Northrop Grumman), Stan Settles (USC), Neil Siegel (Northrop Grumman), and Rich Turner (Stevens) for their valuable contributions to the study.
ICM Practices and Assessment

- From the spiral model to the ICM
  - Principles and example
- Risk-driven incremental definition: ICM Stage I
  - Buying Information to reduce risk
- Risk-driven incremental development: ICM Stage II
  - Achieving both rapid change and high assurance
- Multiple views of the ICM
  - Viewpoints and examples
- ICM Assessment
From the Spiral Model to the ICM

- **Need for Intermediate milestones**
  - Anchor Point Milestones (1996)
- **Avoid stakeholder success model clashes**
  - WinWin Spiral Model (1998)
- **Avoid model misinterpretations**
  - Essentials and variants (2000-2005)
- **Clarify usage in DoD Instruction 5000.2**
  - Initial phased version (2005)
- **Explain system of systems spiral usage to GAO**
  - Underlying spiral principles (2006)
- **Provide framework for human-systems integration**

This chart shows the evolution of the spiral model to accommodate the needs of the software and systems development processes in response to greater system complexities, multiple stakeholders, the ability to rapidly respond to changes, and evolving models and technologies. The latest evolution is the Incremental Commitment Model (ICM). This version has evolved to better handle the integration of human factors, system engineering, and software engineering across the development lifecycle of software-intensive systems.
Process Model Principles

The critical success factor principles were derived from studies of successful and failed projects and the National Research Council’s Human-System Integration study group discussion of particularly important human-system design and development principles. They are not specific to the ICM, but key considerations in any life-cycle process. The primary rationale for each principle is provided below.

1. Without stakeholder commitment and accountability for the system under development, trust and commitment of the development organizations are at risk.

2. A system’s success-critical stakeholders include users, acquirers, developers, maintainers, and potentially others. If stakeholders’ key value propositions are not satisfied by a proposed or delivered system, they will refuse to use it or find ways to undermine it.

3. Single-increment, big-bang system developments take too long to develop and generally produce obsolete, non-responsive systems. Or, they become swamped by change traffic in trying to fix premature commitments.

4,5. Similarly, sequential, non-iterative processes take too long. They also must make premature commitments to poorly-understood requirements, leading to expensive rework or unusable systems.

6. Risk management provides effective ways to prioritize activities and to determine how much of an activity is enough. It also avoids risks turning into expensive or serious problems. Anchor point milestones provide effective ways to synchronize and stabilize concurrent activities.
Incremental Commitment in Gambling

- **Total Commitment: Roulette**
  - Put your chips on a number
    - E.g., a value of a key performance parameter
  - Wait and see if you win or lose
- **Incremental Commitment: Poker, Blackjack**
  - Put some chips in
  - See your cards, some of others' cards
  - Decide whether, how much to commit to proceed

Incremental Commitment in Gambling
A simple metaphor to help understand the ICM is to compare ICM to gambling games such as poker and blackjack, to single-commitment gambling games such as Roulette. Many system development contracts operate like Roulette, in which a full set of requirements is specified up front, the full set of resources is committed to an essentially fixed-price contract, and one waits to see if the bet was a good one or not. With the ICM, one places a smaller bet to see whether the prospects of a win are good or not, and decides to increase the bet based on better information about the prospects of success.
Scalable remotely controlled operations – ICM Case Study

An example to illustrate ICM benefits is the Unmanned Aerial Vehicle (UAV) (or Remotely Piloted Vehicles (RPV)) system enhancement discussed in Chapter 5 of the NRC HSI report [Pew and Mavor, 2007]. The RPVs are airplanes or helicopters operated remotely by humans. These systems are designed to keep humans out of harm’s way. However, the current system is human-intensive, requiring two people to operate a single vehicle. If there is a strong desire to modify the 2:1 (2 people to one vehicle) ratio to allow for a single operator and 4 aircraft (e.g., a 1:4 ratio), based on a proof-of-principle agent-based prototype demo showing 1:4 performance of some RPV tasks, how should one proceed?
Total vs. Incremental Commitment – 4:1 RPV

- **Total Commitment**
  - Agent technology demo and PR: Can do 4:1 for $1B
  - Winning bidder: $800M; PDR in 120 days; 4:1 capability in 40 months
  - PDR: many outstanding risks, undefined interfaces
  - $800M, 40 months: “halfway” through integration and test
  - 1:1 IOC after $3B, 80 months

- **Incremental Commitment [number of competing teams]**
  - $25M, 6 mo. to VCR [4]: may beat 1:2 with agent technology, but not 4:1
  - $75M, 8 mo. to ACR [3]: agent technology may do 1:1; some risks
  - $225M, 10 mo. to DCR [2]: validated architecture, high-risk elements
  - $675M, 18 mo. to IOC [1]: viable 1:1 capability
  - 1:1 IOC after $1B, 42 months

Total vs. Incremental Commitment -- 4:1 RPV

This slide outlines two approaches to the RPV question: total commitment and incremental commitment. While this is a hypothetical case for developing a solution to the RPV manning problem, it shows how a premature total commitment without significant modeling, analysis, and feasibility assessment will often lead to large overruns in costs and schedule, and a manning ratio that is considerably less than initially desired. However, by “buying information” early and validating high-risk elements, the more technologically viable option is identified much earlier and can be provided for a much lower cost and much closer to the desired date. The ICM approach leads to the same improved manning ratio as the total commitment approach, but sooner and at a much reduced cost.

The ICM approach also employs a competitive downselect strategy, which both reduces risk and enables a buildup of trust among the acquirers, developers, and users.
The Cone of Uncertainty: Usual result of total commitment

The Cone of Uncertainty [Boehm, 1981] used empirical data to show the degree to which “if you don’t know exactly what you’re building, you won’t be able to exactly predict its cost.” This view of the cone of uncertainty shows how the total commitment process can lead to competitive bidders making optimistic assumptions about cost. Not only does this set them and the program up for overruns, but it reduces the resources available to perform system architecture and risk resolution by the program’s Preliminary Design Review (PDR).

By “buying information” early about the technology and architecture, this risk can be reduced, thereby significantly reducing the size of the rework and the cone of uncertainty much earlier in the development process.
The ICM is divided into two main stages. Stage I involves incremental system definition. It is divided into three phases, with increasing resource commitments. Sometimes but not always, the increase in commitment will be roughly by a factor of 3, as in the RPV example on chart 40. It had a $25M commitment for its Exploration phase (4 competitors at $5M each, plus $5M for customer evaluation); a $75M commitment for its Validation phase (3 competitors at $20M each, plus $15M for customer evaluation); and a $225M commitment for its Architecting phase (2 competitors at $100M each, plus $25M for customer evaluation). The next few charts elaborate on its content and approach.
The Incremental Commitment Life Cycle Process: Overview

This slide shows how the ICM spans the full life cycle process from concept exploration to operations. Each phase culminates with an anchor point milestone review. At each anchor point, there are 4 options, based on the assessed risk of the proposed system. Some options involve go-backs. These options result in many possible process paths.

The total life cycle is divided into two stages: Stage I of the ICM (Definition) has 3 decision nodes with 4 options/node, culminating with incremental development in Stage II (Development and Operations). Stage II has an additional 2 decision nodes, again with 4 options/node.

One can use ICM risk patterns to generate frequently-used processes with confidence that they fit the situation. Initial risk patterns can generally be determined in the Exploration phase. One then proceeds with development as a proposed plan with risk-based evidence at VCR milestone, adjusting in later phases as necessary.

Risks associated with the system drive the life cycle process. Information about the risk(s) (feasibility assessments) supports the decision to proceed, adjust scope or priorities, or cancel the program.
Anchor Point Feasibility Rationales

- Evidence provided by developer and validated by independent experts that:
  - If the system is built to the specified architecture, it will
  - Satisfy the requirements: capability, interfaces, level of service, and evolution
  - Support the operational concept
  - Be buildable within the budgets and schedules in the plan
  - Generate a viable return on investment
  - Generate satisfactory outcomes for all of the success-critical stakeholders
- All major risks resolved or covered by risk management plans
- Serves as basis for stakeholders’ commitment to proceed

Anchor Point Feasibility Rationales

To make ICM concurrency work, the anchor point milestone reviews are the mechanism by which the many concurrent activities are synchronized, stabilized, and risk-assessed at the end of each phase. Each of these anchor point milestone reviews is focused on developer-produced evidence, documented in a Feasibility Rationale (FR), to help the key stakeholders determine the next level of commitment. At each program milestone/anchor point, feasibility assessments and the associated evidence are reviewed and serve as the basis for the stakeholders’ commitment to proceed.

The FR is not just a document, a set of PowerPoint charts, or Unified Modeling Language (UML) diagrams. It is based on evidence from simulations, models, or experiments with planned technologies and detailed analysis of development approaches and projected productivity rates. The detailed analysis is often based on historical data showing reuse realizations, software size estimation accuracy, and actual developer productivity rates.

It is often not possible to fully resolve all risks at a given point in the development cycle, but known, unresolved risks need to be identified and covered by risk management plans.
There is Another Cone of Uncertainty: Shorter increments are better

Uncertainties in competition and technology evolution and changes in organizations and mission priorities, can wreak havoc with the best of system development programs. In addition, the longer the development cycle, the more likely it will be that several of these uncertainties or changes will occur and make the originally-defined system obsolete. Therefore, planning to develop a system using short increments helps to ensure that early, high priority capabilities can be developed and fielded and changes can be more easily accommodated in future increments.
The Incremental Commitment Life Cycle Process: More on the Overview

Stage II of the Incremental Commitment Life Cycle provides a framework for concurrent engineering and development of multiple increments. More on this concurrency follows on the next slides.

Note: The term “concurrent engineering” fell into disfavor when behind-schedule developers applied it to the practice of proceeding into development while the designers worked on finishing the design. Not surprisingly, the developers encountered the rework penalty described on chart 17 for going into development with weak architecture and risk resolution.

“Concurrent engineering” as applied in the ICM is much different. It is focused on doing a cost-effective job of architecture and risk resolution in Stage I; and on performing stabilized development, verification, and validation of the current system increment while concurrently handling the systems change traffic and preparing a feasibility-validated architecture and set of plans for the next increment in Stage II.
ICM Stage II: Increment View

The ICM is organized to simultaneously address the conflicting challenges of rapid change and high assurance of dependability. It also addresses the need for rapid fielding of incremental capabilities with a minimum of rework.

For high assurance, the development of each increment should be short, stable, and provided with a validated baseline architecture and set of requirements and development plans. The architecture should accommodate any foreseeable changes in the requirements; the next chart shows how the unforeseeable changes are handled.
ICM Stage II: Increment View

A radical idea?

No; a commercial best practice and part of DoDI 5000.2

12/31/2007

©USC-CSSE

ICM Stage II: More Detailed Increment View

The need to deliver high-assurance incremental capabilities on short fixed schedules means that each increment needs to be kept as stable as possible. This is particularly the case for large, complex systems and systems of systems, in which a high level of rebaselining traffic can easily lead to chaos. In keeping with the use of the spiral model as a risk-driven process model generator, the risks of destabilizing the development process make this portion of the project into a waterfall-like build-to-specification subset of the spiral model activities. The need for high assurance of each increment also makes it cost-effective to invest in a team of appropriately skilled personnel to continuously verify and validate the increment as it is being developed.

However, “deferring the change traffic” does not imply deferring its change impact analysis, change negotiation, and rebaselining until the beginning of the next increment. With a single development team and rapid rates of change, this would require a team optimized to develop to stable plans and specifications to spend much of the next increment’s scarce calendar time performing tasks much better suited to agile teams.

The appropriate metaphor for addressing rapid change is not a build-to-specification metaphor or a purchasing-agent metaphor but an adaptive “command-control-intelligence-surveillance-reconnaissance” (C2ISR) metaphor. It involves an agile team performing the first three activities of the C2ISR “Observe, Orient, Decide, Act” (OODA) loop for the next increments, while the plan-driven development team is performing the “Act” activity for the current increment. “Observing” involves monitoring changes in relevant technology and COTS products, in the competitive marketplace, in external interoperating systems and in the environment; and monitoring progress on the current increment to identify slowdowns and likely scope deferrals. “Orienting” involves performing change impact analyses, risk analyses, and tradeoff analyses to assess candidate rebaselining options for the upcoming increments. “Deciding” involves stakeholder renegotiation of the content of upcoming increments, architecture rebaselining, and the degree of COTS upgrading to be done to prepare for the next increment. It also involves updating the future increments’ Feasibility Rationales to ensure that their renegotiated scopes and solutions can be achieved within their budgets and schedules.

A successful rebaseline means that the plan-driven development team can hit the ground running at the beginning of the “Act” phase of developing the next increment, and the agile team can hit the ground running on rebaselining definitions of the increments beyond.
As with complex architectures needing multiple views to be understood by multiple stakeholders, so with complex processes. The next few charts provide several such views of the ICM.
MBASE-RUP/ICM Anchor Points Enable Concurrent Engineering

This slide illustrates the Rational Unified Process (RUP) and its anchor points. The RUP activities and anchor points were developed to support concurrent engineering and are the basis for the ICM anchor points. In comparison to the software-intensive RUP, the ICM also addresses hardware and human factors integration. It extends the RUP phases to cover the full system life cycle: an Exploration phase precedes the RUP Inception phase, which is refocused on valuation and investment analysis. The RUP Elaboration phase is refocused on System Architecting (a term based on the [Rechtin, 1991] analogy of system architecting to the architecting of buildings, involving concurrent development of requirements, architecture, and plans, to which it adds feasibility evidence); the RUP Construction and Transition phases are combined into Development; and an additional Operations phase combines operations, production, maintenance, and phase-out.
ICM HSI Levels of Activity for Complex Systems

As mentioned earlier, with the ICM, a number of system aspects are being concurrently engineered at an increasing level of understanding, definition, and development. The most significant of these aspects are shown in this slide, an extension of a similar view of concurrently engineered software projects developed as part of the RUP (shown on the previous slide).

As with the RUP version, it should be emphasized that the magnitude and shape of the levels of effort will be risk-driven and likely to vary from project to project. In particular, they are likely to have mini risk/opportunity-driven peaks and valleys, rather than the smooth curves shown for simplicity in this slide. The main intent of this view is to emphasize the necessary concurrency of the primary success-critical activities shown as rows. Thus, in interpreting the Exploration column, although system scoping is the primary objective of the Exploration phase, doing it well involves a considerable amount of activity in understanding needs, envisioning opportunities, identifying and reconciling stakeholder goals and objectives, architecting solutions, life cycle planning, evaluation of alternatives, and negotiation of stakeholder commitments.
Different Risk Patterns Yield Different Processes

As illustrated in the four example paths through the Incremental Commitment Model in this slide, the ICM is not a single monolithic one-size-fits-all process model. As with the spiral model, it is a risk-driven process model generator, but the ICM makes it easier to visualize how different risks create different processes.

In Example A, a simple business application based on an appropriately-selected Enterprise Resource Planning (ERP) package, there is no need for a Valuation or Architecting activity if there is no risk that the ERP package and its architecture will not cost-effectively support the application. Thus, one could go directly into the Development phase, using an agile method such as a Scrum/Extreme Programming combination would be a good fit. There is no need for Big Design Up Front (BDUF) activities or artifacts because an appropriate architecture is already present in the ERP package. Nor is there a need for heavyweight waterfall or V-model specifications and document reviews. The fact that the risk at the end of the Exploration phase is negligible implies that sufficient risk resolution of the ERP package’s human interface has been done.

Example B involves the upgrade of several incompatible legacy applications into a service-oriented web-based system. Here, one could use a sequential waterfall or V-model if the upgrade requirements were stable, and its risks were low. However, if for example the legacy applications’ user interfaces were incompatible with each other and with web-based operations, a concurrent risk-driven spiral, waterfall, or V-model that develops and exercise extensive user interface prototypes and generates a Feasibility Rationale (described on chart 44) would be preferable.

In Example C, the stakeholders may have found during the Valuation phase that their original assumptions about the stakeholders having a clear, shared vision and compatible goals with respect the proposed new system’s concept of operation and its operational roles and responsibilities were optimistic. In such a case, it is better to go back and assure stakeholder value proposition compatibility and feasibility before proceeding, as indicated by the arrow back into the valuation phase.

In Example D, it is discovered before entering the Development phase that a superior product has already entered the marketplace, leaving the current product with an infeasible business case. Here, unless a viable business case can be made by adjusting the project’s scope, it is best to discontinue it. It is worth pointing out that it is not necessary to proceed to the next major milestone before terminating a clearly non-viable project, although stakeholder concurrence in termination is essential.
Common Risk-Driven Special Cases of the ICM

As mentioned before, as the ICM is used, different risk patterns can be identified, leading to the development of special cases of the ICM to handle these risk patterns. This slide highlights some of the more common risk patterns, describes their characteristics, and explains how the ICM can be most effectively used to develop this type of system.

For each familiar risk pattern, the table identifies the counterpart familiar life cycle process pattern, along with a representative example project type, the project’s key Stage I and Stage II activities, and a rough estimate of the length of its internal builds and deliverable increments. In general, the risk patterns will be well enough established during the ICM Exploration phase to enable the project to determine and switch to the appropriate familiar process without having to go thorough the full generality of the ICM framework.

For example, Row 1 indicates that if a small accounting application can be completely supported by an NDI package, its life cycle process is to acquire and tailor the NDI package. Row 2 corresponds to the agile project described as Example A in the previous chart.

The following slides elaborate on two more of these special cases.
Several further references are TBD.
References -II

Lane, J. and Boehm, B., "System of Systems Cost Estimation: Analysis of Lead System Integrator
Madacy, R., Boehm, B., Lane, J., "Assessing Hybrid Incremental Processes for SEIS Development", USC
Maier, M., "System and Software Architecture Reconciliation," Systems Engineering 9 (2), 2006, pp. 146-
159.
Engineering Institute, 2006.
Pew, R. W., and Mavor, A. S., Human-System Integration in the System Development Process: A New Look,
Schorr, T., "Integrating Systems and Software Engineering: Observations in Practice," OSD/USC
Integrating Systems and Software Engineering Workshop,
http://csse.usc.edu/events/00/sr-00 ┬┌∑ sistem program nam, nctober 2001.
SoSE-Related References


Dillmar, Mike (2006); "System of Systems Characteristics and Interoperability in Joint Command Control", Proceedings of the 2nd Annual System of Systems Engineering Conference

Elaboration Industries Alliance (1999), EIA Standard 632, "Process for Engineering a System

Finley, James (2005); "Keynote Address", Proceedings of the 2nd Annual System of Systems Engineering Conference

Garber, Vital (2005); "Keynote Presentation", Proceedings of the 2nd Annual System of Systems Engineering Conference


Kygiel, A. (1999); Behind the Wizard’s Curtain; CCRP Publication Series, July, 1999, p. 33


Melich, Abe (2009); "System of Systems Engineering (SoSE) and Architecture Challenges in a Net Centric Environment", Proceedings of the 2nd Annual System of Systems Engineering Conference

Pair, Major General Carlos (2005); "Keynote Presentation", Proceedings of the 2nd Annual System of Systems Engineering Conference

Proceedings of AFOSR SoSE Workshop, Sponsored by Purdue University, 17-18 May 2006


Slad, Carl (2006); "Keynote Presentation", Proceedings of the 2nd Annual System of Systems Engineering Conference


12/31/2007
Several further acronyms are TBD

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACR</td>
<td>Architecting Commitment Review</td>
</tr>
<tr>
<td>B/L</td>
<td>Baselined</td>
</tr>
<tr>
<td>CCD</td>
<td>Core Capability Drive-Through</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial Off-the-Shelf</td>
</tr>
<tr>
<td>DCR</td>
<td>Development Commitment Review</td>
</tr>
<tr>
<td>DI</td>
<td>Development Increment</td>
</tr>
<tr>
<td>DUOMLF</td>
<td>Doctrine, Organization, Training, Materiel, Leadership, Personnel, Facilities</td>
</tr>
<tr>
<td>ECR</td>
<td>Exploration Commitment Review</td>
</tr>
<tr>
<td>FMEA</td>
<td>Failure Modes and Effects Analysis</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HSI</td>
<td>Human-System Interface</td>
</tr>
<tr>
<td>ICM</td>
<td>Incremental Commitment Model</td>
</tr>
<tr>
<td>IOC</td>
<td>Initial Operational Capability</td>
</tr>
<tr>
<td>IRR</td>
<td>Inception Readiness Review</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>LCA</td>
<td>Life Cycle Architecture</td>
</tr>
<tr>
<td>LCO</td>
<td>Life Cycle Objectives</td>
</tr>
<tr>
<td>OC</td>
<td>Operational Capability</td>
</tr>
<tr>
<td>OCR</td>
<td>Operations Commitment Review</td>
</tr>
<tr>
<td>OO&amp;D</td>
<td>Observe, Orient and Decide</td>
</tr>
<tr>
<td>OODA</td>
<td>Observe, Orient, Decide, Act</td>
</tr>
<tr>
<td>O&amp;M</td>
<td>Operations and Maintenance</td>
</tr>
<tr>
<td>PRR</td>
<td>Product Release Review</td>
</tr>
<tr>
<td>RACRS</td>
<td>Regional Area Crisis Response System</td>
</tr>
<tr>
<td>SoS</td>
<td>System of Systems</td>
</tr>
<tr>
<td>SoSE</td>
<td>System of Systems Engineering</td>
</tr>
<tr>
<td>TSE</td>
<td>Traditional Systems Engineering</td>
</tr>
<tr>
<td>VCR</td>
<td>Validation Commitment Review</td>
</tr>
<tr>
<td>V&amp;V</td>
<td>Verification and Validation</td>
</tr>
</tbody>
</table>