Software and IT Cost IPT Proceedings

August 10-12, 2015

Lockheed Martin Global Vision Center,
2121 Crystal Drive
Crystal City, Arlington, VA 22202

COMMITTEE

Chair:
Wilson Rosa (NCCA)

Planning Co-Chairs
Shelley Dickson (NCCA)
Corey Boone (NCCA)
John Murray (NCCA)
Richard Mabe (AFCAA)

Program Committee:
Cheryl Jones (Us Army ARDEC)
Garry Roedler (Lockheed Martin)
Nieman Gregory (Lockheed Martin)

Venue:
Garry Roedler (Lockheed Martin)
Nieman Gregory (Lockheed Martin)

TABLE OF CONTENTS

OVERVIEW

The Naval Center for Cost Analysis will hold the inaugural Software and Information Technology Cost Integrated Product Team (Software and IT Cost IPT) meeting from August 10-12, 2015 at the Lockheed Martin Global Vision Center in Crystal City, Virginia. This meeting is organized with the support of US Army ARDEC, Lockheed Martin, and DoD cost agencies.

The Software and IT Cost IPT meeting is a venue to build coalitions with government and industry, to exchange cost data, share lessons learned, and establish best practices concerning software and information technology cost estimation. Topics of interest include:

- Software cost estimation
- Software schedule estimation
- Information Technology (IT) cost estimation
- Cost Data Collection and Analysis Best Practices
- Functional size measurements
- Early phase software and IT cost estimation
- IT Cost Measures and Benchmarks
- Measurements for agile or other developmental approaches
- Measurements for software maintenance and sustainment
- Measurements for cloud computing services – SaaS, PaaS, IaaS
- Measurements for IT help desk and support
- Measurements for data center and network consolidation

The program includes presentations, workshops, and contractor one-on-one discussions. Presentations and workshops are opened to all attendees. Contractor one-on-one discussions are restricted to federal employees who have registered. For more details regarding the program and venue please review the attached brochure.

BIOGRAPHIES

Opening Remarks and Keynote

Dr. Richard P. Burke, Deputy Director, Cost Assessment

Office of the Secretary of Defense/Cost Assessment and Program Evaluation

Dr. Burke has served as the Deputy Director, Cost Assessment, in the Office of the Secretary of Defense, Cost Assessment and Program Evaluation (OSD/CAPE) since June, 2009. Prior to this he served as the Deputy Director for Resource Analysis, Office of the Secretary of Defense, Program Analysis and Evaluation, and as the Chairman of the Cost Analysis Improvement Group (CAIG) beginning in November 2002. He joined the Office of the Secretary of Defense in April 1988 prior to his service in DoD.

Dr. Burke served in several program management positions at Sandia National Laboratories in Albuquerque, New Mexico. He is an International Affairs Fellow of the Council on Foreign Relations in New York, and served as a visiting scholar at Stanford University during 1992-93. Educated at the Massachusetts Institute of Technology, he received a doctorate in nuclear engineering and decision analysis in 1984. His published work includes studies of the economic and international aspects of commercial nuclear power reactors, the economic risks of power reactor accidents, and export controls on high-technology industries.

Session 1: GAO Scheduling: Best Practices in an Agile Environment

Karen A Richey, Assistant Director for the Applied Research and Methods Team, Government Accountability Office (GAO)

Ms. Karen Richey is an Assistant Director for the Applied Research and Methods Team at the Government Accountability Office (GAO). She is a senior cost analyst responsible for performing cost, schedule, and Earned Value Management analyses to support audit findings on a wide range of government programs. Certified by the National Defense University as a Chief Information Officer (CIO), Karen is also Level-III certified in the field of cost estimating and financial management. She has 22 years experience in the fields of cost estimating, scheduling, and EVM analysis.

Before joining the GAO, Karen was a cost analyst for the Department of the Navy where she performed earned value management and developed independent cost estimates for major weapon and automated information systems. Karen holds a degree in Statistics and Mathematics from the University of South Carolina. She led the development of the GAO Cost Estimating and Assessment Guide which provides auditors with best practice criteria for determining the reliability of cost estimates. In addition to auditing, Karen also provides training classes for auditors on cost estimating, schedule analysis, and EVM.

Session 2: Automated Information Systems (AIS) Cost Estimating Method Matrix

Richard Mabe, Technical Advisor, Air Force Cost Analysis Agency

Mr. Mabe is the Technical Advisor for the IT and Electronics Division of the Air Force Cost Analysis Agency. In this role, Mr. Mabe supports cost estimating and analysis for all Air Force MAIS programs, as well as major C4ISR system MDAP programs. He leads research efforts to identify data and develop estimating methodologies for a wide variety of IT and electronic systems and equipment, to include cyber security, and provides subject matter expertise to other service and DOD level working groups.

Mr. Mabe has a BS from Boise State University in Geology and an MS from AFIT in Logistics. He has worked continuously within the DOD as an AF Officer, a contractor and a civil servant since 1977.

Session 3: Inherent Issues of Agile EVM and A Viable Solution for Effective Agile Project Controls

Omar Mahmoud, Barakah Consulting Group, Inc.

As President of Barakah Consulting Group, Inc., Mr. Mahmoud has 15 years of Analytics, Finance, and software development experience supporting private and public sector clients. Mr. Mahmoud holds a Bachelor in Computer Science, an MBA, is CCE/A and Scrum Master certified. He currently supports PG&E's cyber-security protection project and has presented at several ICEAA conferences on subjects related to Agile software baselining and project controls, software cost estimation, PLCCE Design and Modeling, and Economic Analysis.

Session 4: Measuring Benefits of Innovation

Steve Shyman, Boeing

Mr. Steve Shyman, Estimating Specialist, over 30 years in the aerospace industry involved in software engineering, estimating, planning, measurement, risk assessment, affordability, and data collection and analysis. Steve has worked both the commercial and defense sides of Boeing in both the engineering and finance areas. In addition to software engineering Steve also supports Systems Engineering estimating and affordability, and leads an Integrated Estimating/Engineering Working Group.

Shawn Rahmani, Boeing

Dr. Shawn Rahmani is a Senior Technical Fellow at the Boeing Defense, Space and Security, specialized in Software Engineering. He has over 35 years of experience in the areas of avionics and mission critical software systems. Shawn leads the Systems and Software Development and Integration Core Technology for Boeing that includes process and tools to improve software affordability and project execution.

Session 5: COSYSMO 3.0: Updating Cost Estimation of Systems Engineering to Support Affordability

Jim Alstad, University of Southern California

Jim Alstad recently retired from a 32-year career at Boeing, where he led satellite flight software projects. His main interest now is pursuing a PhD under Dr Barry Boehm at USC. His thesis topic is COSYSMO 3.0, a unification of previous developments in estimating models for systems engineering costs. He is also advising on the development of the COCOMO III software cost estimating model.

Dr. Barry Boehm, University of Southern California

Dr. Boehm is an American software engineer, TRW Emeritus Professor of Software Engineering at the Computer Science Department of the University of Southern California, and known for his many contributions to software engineering.

Barry Boehm received his B.A. degree from Harvard in 1957 and his M.S. and Ph.D. degrees from UCLA in 1961 and 1964, all in Mathematics. He also received an honorary Sc.D. in Computer Science from the U. of Massachusetts in 2000.

Session 6: Estimating Agile Software Cost for Iterative/Incremental Development Programs

Dr. Bob Hunt, Galorath, Inc.

Dr. Bob Hunt has served in various technical and senior management positions in industry including Vice President and Operations Manager at SAIC, Vice President and Board of Directors of CALIBRE Systems, President CALIBRE Services, Vice President for Services at Galorath Incorporated, President of Dulos Inc., and Board of Directors for ALATEC, Inc

Session 7: How Much Does Software Maintenance Cost?

Cheryl Jones, US Army ARDEC

Ms. Jones is a lead software engineer in the Quality Engineering & System Assurance Sciences Group at RDECOM-ARDEC at Picatinny, New Jersey. She is responsible for measurement and analysis, risk management, cost estimation, and decision analysis technology development and application across a wide base of DoD, Government, and Commercial programs and organizations. Ms. Jones is the DoD representative to the International Standards Organization SC7, System and Software Engineering.

Session 8: Challenges with Sizing and Estimating Enterprise Information Systems

Dr. Chris Miller, QSM, Inc.

David Fersch, ODASA-CE

Session 9: DoD Enterprise Software Initiative

Floyd Groce, Department of Navy, Chief Information Officer

Mr. Floyd Groce joined the DON CIO staff in 1998. He is the Co-Chair of the DoD ESI Working Group. He leads the DON CIO Enterprise Licensing and strategic sourcing efforts for IT hardware, software and services and is one of the DoD Subject Matter Experts for the Office of Management and Budget Software Category Management initiative under the OMB-chartered Enterprise Software Category Team. Mr. Groce holds a bachelor’s degree in Business Administration from Dakota State University.

Session 10: The Software Metrics Cube: Software Acquisition Simplified for Government Oversight Offices

Victor Fuster, QSM, Inc.

Victor Fuster is a Senior Manager at Quantitative Software Management (QSM) Inc. in McLean, VA and has over 15 years of experience providing DoD software estimation and technology protection. He's supported notable clients across DoD and federal government including the Office of the Secretary of Defense (OSD), Office of Cost Analysis & Program Assessment (CAPE) and Missile Defense Agency (MDA). He received his M.S. from Eastern Kentucky University and is a US Marine Corps veteran.

Session 11: Why are Estimates Always Wrong: Estimation Bias

Dan Galorath, CEO, Galorath, Inc.

Mr. Galorath has been involved with engineering, estimating and analysis for 40 years. Under his leadership Galorath Incorporated, the developed SEER cost, schedule, risk applications, methods, and training for software, hardware, electronics & systems, Information Technology, and Manufacturing. His widely read blog, Dan Galorath on estimating, covers estimation, planning, measurement, control and risk analysis. He has received lifetime achievement awards from SCEA and ISPA.

Session 12: COCOMO III Overview

Bradford Clark, Vice-President, Software Metrics Inc.

Dr. Brad Clark is Vice-President of Software Metrics Inc. – a Virginia based consulting company. His area of expertise is in software cost and schedule data collection, analysis and parametric modeling.

Dr. Clark received a Ph.D. in Computer Science in 1997 from the University of Southern California. He co-authored a book with Barry Boehm and others on "Software Cost Estimation with COCOMO II." Brad is a former US Navy A-6 Intruder pilot.

Lockheed Martin Global Vision Center
2121 Crystal Drive
Arlington, VA 22202

Software and IT Cost IPT Agenda

10-12 August 2015

Monday, August 10, 2015 – Contractor Discussions (Restricted)

0830 – 0900	Registration		
0900 – 1130	**Amazon Web Services** One-on-One	Mr. Richard Mabe (AFCAA) Mr. Nate Mangan (Amazon)	First Floor Auditorium
1130 – 1300	Lunch		
1300 – 1445	**Northrup Grumman** One-on-One	Dr. Wilson Rosa (NCCA) Mr. John Sautter (Northrop Grumman)	First Floor Auditorium
1445 – 1500	Break		
1500 – 1630	**BAE Systems** One-on-One	Dr. Wilson Rosa (NCCA) Dr. Gan Wang (BAE Systems)	First Floor Auditorium

Tuesday, August 11, 2015 – General Session (Open to All)

0800 – 0830	**Registration**		
0830 – 0840	Introduction	Dr. Wilson Rosa (NCCA)	First Floor Auditorium
0840 – 0900	Opening Remarks and Keynote	Dr. Richard P Burke (OSD CAPE)	
0900 – 0930	GAO Scheduling: *Best Practices in an Agile Environment*	Ms. Karen Richey (GAO)	
0930 – 1000	*AIS Cost Estimating Method Matrix*	Mr. Richard Mabe (AFCAA)	
1000 – 1015	**Break**		
1015 – 1045	*Inherent Issues of Agile EVM and Viable Solutions for Effective Agile Project Controls*	Mr. Omar Mahmoud (Barakah Consulting Group.)	First Floor Auditorium
1045 – 1115	*Measuring Benefits of Innovation*	Mr. Steve Shyman and Dr. Shawn Rahmani (Boeing)	
1115 – 1145	*COSYSMO 3.0: Updating Cost Estimation of Systems Engineering to Support Affordability*	Mr. Jim Alstad (USC) Dr. Barry Boehm (USC)	
1145 – 1300	**Lunch**		
1300 – 1630	**COSYSMO 3.0** Workshop	Mr. Jim Alstad (USC) Dr. Barry Boehm (USC)	Second Floor GVC-A

Lockheed Martin Global Vision Center
2121 Crystal Drive
Arlington, VA 22202

Software and IT Cost IPT Agenda

10-12 August 2015

Tuesday, August 11, 2015 – Contractor Discussions (Restricted)			
1300 – 1430	**Raytheon** One-on-One	Dr. Corinne Wallshein (NCCA) Ms. Cynthia Rogers (Raytheon)	Second Floor GVC-B
Wednesday, August 12, 2015 – General Session (Open to All)			
0800– 0820	**Registration**		
0820 – 0830	Introduction	Dr. Wilson Rosa (NCCA)	First Floor Auditorium
0830 – 0900	*Estimating Agile Software Cost for Iterative/ Incremental Development Programs*	Dr. Bob Hunt (ODASA/Galorath)	
0900 – 0930	*What Does Software Maintenance Cost?*	Ms. Cheryl Jones (ODASA)	
0930 – 1000	*Challenges with Sizing and Estimating Enterprise Info Systems*	Dr. Chris Miller (QSM / ODASA) Mr. David Fersch (ODASA)	
1000 – 1015	**Break**		
1015 – 1045	*DOD Enterprise Software Initiative*	Mr. Floyd Groce (DON CIO)	First Floor Auditorium
1045 – 1115	*The Software Metrics Cube: Software Acquisition Simplified for Government Oversight Offices*	Mr. Victor Fuster (QSM)	
1115 – 1145	*Why Can't People Estimate: Estimating Bias*	Mr. Dan Galorath (Galorath)	
1145 – 1215	**COCOMO III** Workshop Overview	Dr. Brad Clark (USC)	
1215 – 1315	**Lunch**		
1315 – 1645	**COCOMO III** Workshop	Dr. Brad Clark (USC)	Second Floor GVC-A

Keynote Speaker

Dr. Richard P. Burke

Deputy Director, Cost Assessment

Office of the Secretary of Defense, Cost Assessment and Program Evaluation

Software and Information Technology Cost Integrated Product Team (IPT)

Dr. Wilson Rosa
Cost IPT Chairperson

August 11, 2015

Executive Committee

Chairperson	• Wilson Rosa (NCCA)
Co-Chairs	• Shelley Dickson (NCCA) • Corey Boone (NCCA) • John Murray (NCCA)
Program Committee	• Cheryl Jones (ARDEC) • Garry Roedler (Lockheed Martin) • Greg Niemann (Lockheed Martin) • Richard Mabe (AFCAA)

Mission Statement

- Build coalitions with government, industry, academia to exchange cost data, lessons learned, best practices concerning Software and Information Technology cost estimation

Goals

- Augment cost data reporting practices and policies for Major Automated Information Systems
- Standardize software cost data definitions reported in Contract Data Reporting Listing (CDRL) requirements
- Improve ability to efficiently host, share and request contractor data between Government agencies

Focus Areas

Software

- ❑ Cost Estimation Best Practices
- ❑ Schedule Estimation Measures
- ❑ Early Phase Agile Cost Measures
- ❑ Early Phase Size Measurements
- ❑ Quantifying Cyber Security Requirements and Cost Measures
- ❑ COTS Integration Cost Measures
- ❑ Data Collection Best Practices
- ❑ Open Source Cost Models

Information Technology

- ❑ IT Cost Measures and Benchmarks for
  - ✓ Enterprise Resource Planning
  - ✓ Early phase IT Implementations
  - ✓ Cloud services – SaaS, PaaS, IaaS
  - ✓ Help Desk
  - ✓ System Administration
  - ✓ Data Center Consolidation
  - ✓ Network Consolidation
  - ✓ Data Cleansing
  - ✓ Data Migration
- ❑ IT Data Collection Best Practices
- ❑ Early Phase Cost Measures
- ❑ Acquisition and Contract Strategies

What's in it for YOU?

- Cost data sharing among contractor and government sources
- Exploit opportunities to engage and potential for substantive mutual areas for improvement
- Collaborate with Industry and Academia for the development of open-source Cost Estimating Relationships, benchmarks, etc.

GAO

GAO Scheduling Best Practices Applied to an Agile Setting

August 11, 2015

Karen Richey

Outline

- Why Scheduling?
- GAO Schedule Assessment Guide Overview
- GAO Schedule Assessment Guide Status
- Agile Appendix Overview
- Scheduling 10 Best Practices and Agile

Why Scheduling?

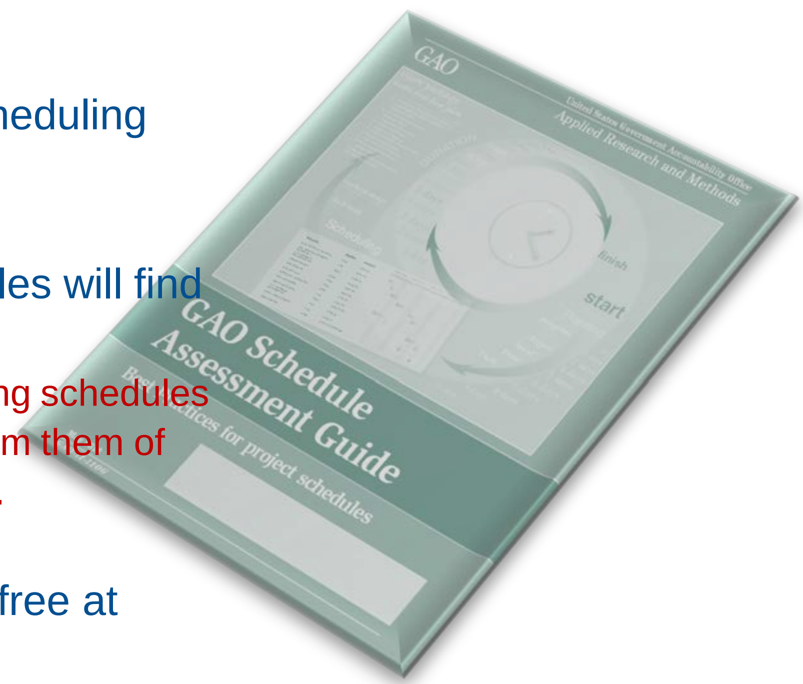
- A reliable schedule is necessary for successful program management

- Developing an integrated schedule is key for managing program performance and is necessary for determining what work remains and the expected cost to complete it.

- The success of any program, therefore, depends in part on having a reliable schedule

GAO Schedule Assessment Guide Overview

- The GAO Schedule Assessment Guide was published in draft form in May 2012
- The GAO Schedule guide further develops the scheduling concepts introduced in the GAO Cost guide.
- The project team that develops a project's schedules will find the guide indispensable.
    - Agencies that do not have a formal policy for creating schedules will benefit from using the guide because it will inform them of GAO's criteria for assessing a schedule's credibility.
- The GAO Schedule guide can be downloaded for free at [www.**gao**.gov/products/**GAO**-12-120G](www.gao.gov/products/GAO-12-120G)



Schedule Assessment Guide Status

- Development of Exposure Draft (Nov 2010 – May 2012)
- Development of Final Draft (May 2012 – September 2015 expected)
- Reviewing organizations span private industry (80), government departments/agencies (29), and trade groups/universities (4).
- Final draft will include updated figures, schedule risk analysis, and an appendix devoted to scheduling in an Agile Development Environment

Agile Appendix Overview

- Many GAO audits for IT systems did not have schedules because they were using the Agile method to develop software

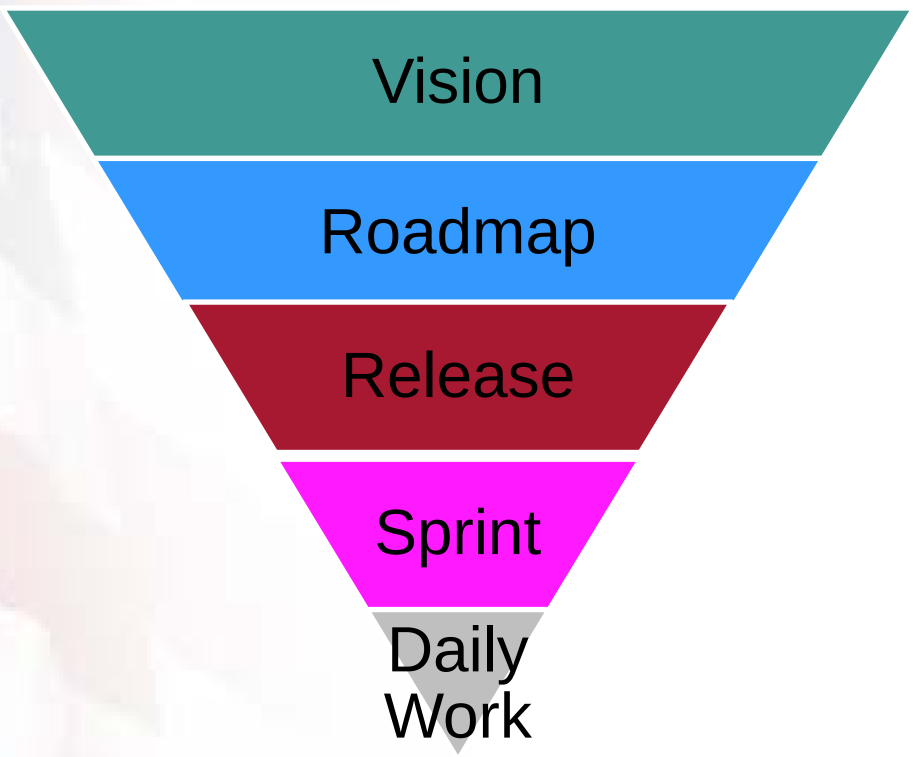
- Purpose of the appendix is to identify common misconceptions about scheduling for Agile projects and dispel them

Key Point: Agile measures progress through the implementation of working software

GAO Scheduling Best Practices

- Our research has identified ten best practices associated with developing maintaining a reliable schedule:
  1. Capturing all activities
  2. Sequencing all activities
  3. Assigning resources to all activities
  4. Establishing the duration of all activities
  5. Verifying that the schedule can be traced horizontally and vertically
  6. Confirming that the critical path is valid
  7. Ensuring reasonable total float
  8. Conducting a schedule risk analysis
  9. Updating the schedule using actual progress and logic
  10. Maintaining a Baseline Schedule (new)

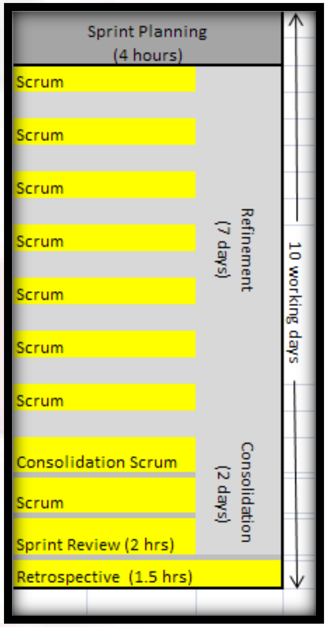
Best Practice 1: Capture All Activities



- Roadmap with prioritized must have features developed with input from stakeholders & SMEs
- Schedule reflecting all activities the government, its contractors, and other need to perform to deliver must have requirements
  - Lowest planning level included in the IMS will be the Release level with the understanding that a specific number of sprints will be planned for each release
- Roadmap linkage to SOW
- Prioritized product backlog consisting of epics, features, and stories
  - Product backlog queues unfinished work and any defects in priority order

Best Practice 2: Sequencing All Activities

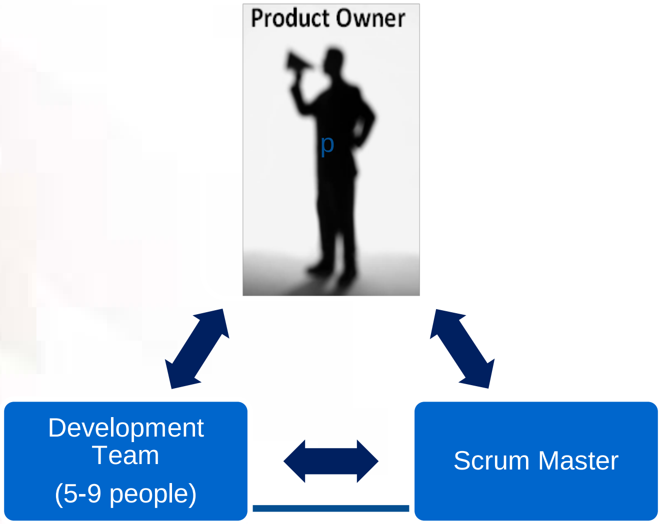
- In an Agile setting time is fixed with a steady team to complete the work so tracking sprints in a schedule means you are only monitoring level of effort and you will not know the true progress

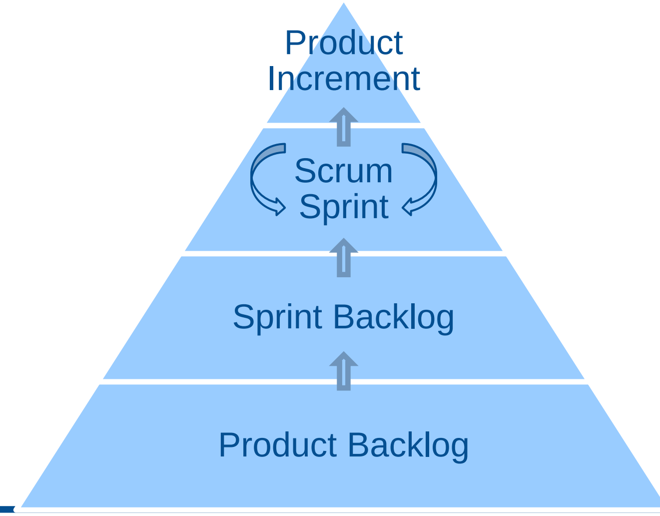


- Better to use Agile metrics (e.g., daily standup meetings, completed stories, burndown charts, etc.) to determine the amount of working software scope being delivered in each sprint
- Product and sprint backlogs rank requirements in order of priority
- Progress board shows sequencing of work within a sprint
- Dependencies for sprints should be reflected in the schedule at the release level

Best Practice 3: Assigning Resources to All Activities

- Does not apply to Agile sprints since the development team is usually stable, only the scope will be variable
  - Team size should be small (between 5-9 people); there can be several sprint teams working in parallel
- Scrum master coaching by qualified experts
- Organizational training in the Agile method
- Progress board shows resources that are working on stories in a sprint





Best Practice 4: Establishing the Duration of All Activities

- Consistent sprint durations of 2-4 weeks
- Release durations are tracked in the IMS and based on the number of sprints planned to deliver must have features
  - Releases will often have long durations (> 2 months)
- Velocity is tracked to measure the number of story points implemented per sprint
- Sprints consist of the following steps: planning, coding, testing, demonstrating working software to the customer, and conducting a retrospective

Best Practice 5: Verifying the Schedule can be Traced Horizontally and Vertically

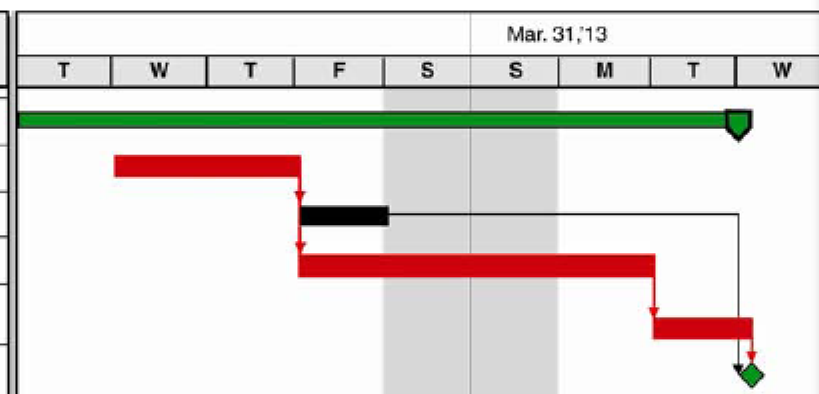
- This best practice only applies to the activities in the schedule down to the release level
  - **<u>Horizontal Integration</u>**: Below the release level, horizontal integration can be determined by examining whiteboards which show using sticky notes or index cards what work has been done, what work is underway, and what work is still left to be accomplished in a sprint.
    - Releases are included in the program schedule along with any dependencies
  - **<u>Vertical Integration</u>**: Below the release level, the reliance on Agile metrics like the burn down chart will provide management and oversight officials information on what work is done and how this corresponds to work status in the release
    - Roadmap shows high level plan for implementing must have features that should trace down to product and sprint backlogs

Best Practice 6: Verifying that the Critical Path is Valid

- The schedule should reflect the sequence of releases that identify must haves features so that a critical path can be identified and tracked
  - There will be no critical path below the release level
  - Instead the reliance on Agile metrics is necessary for determining what features and user stories can be delivered in each sprint cycle
  - At the sprint level, the critical path is managed during sprint planning and daily stand up meetings

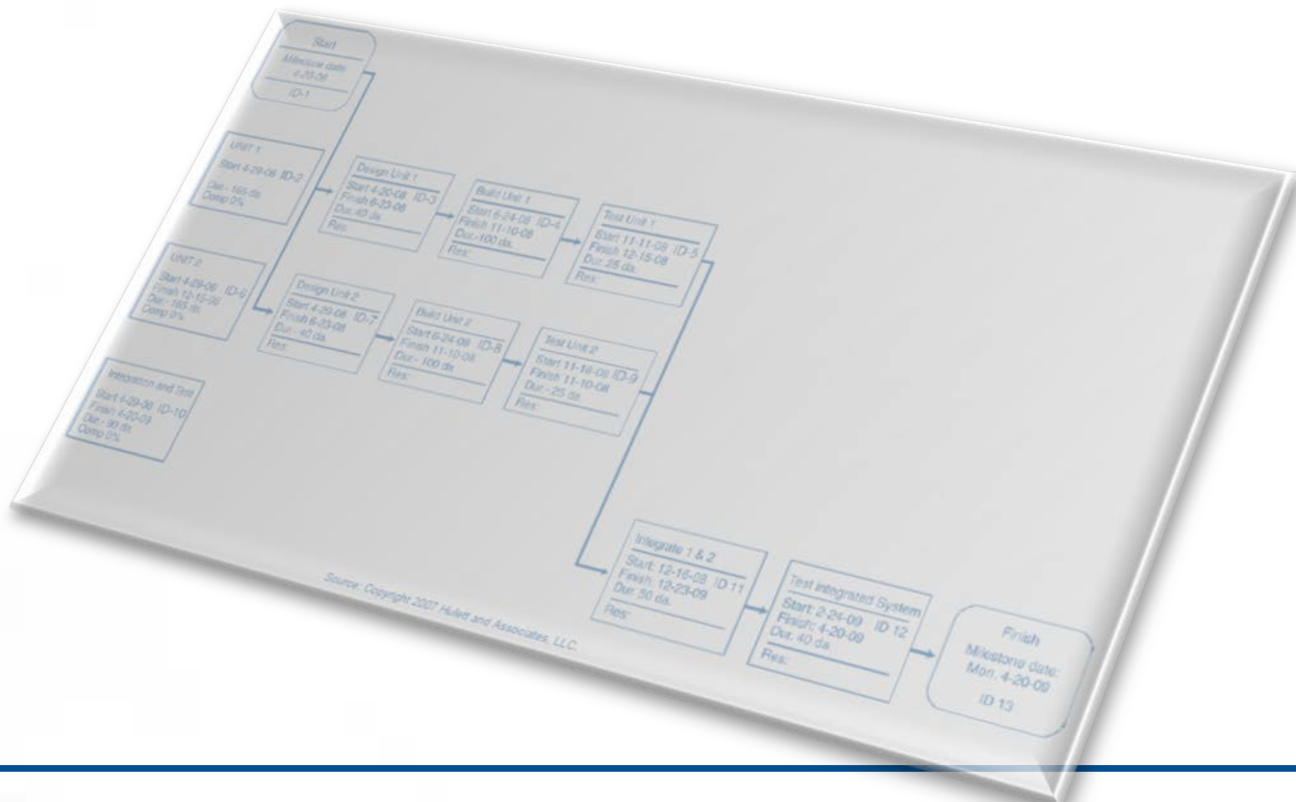
Figure 25: The Critical Path and Total Float

Activity name	Early start	Late start	Total float (in days)
Framing	3-11-13	3-11-13	0
Install roof decking	3-27-13	3-27-13	0
Rough-in framing inspection	3-29-13	4-2-13	2
Form and pour driveway	3-29-13	3-29-13	0
Finish excavate and pour garage floor	4-2-13	4-2-13	0
Framing complete	4-2-13	4-2-13	0

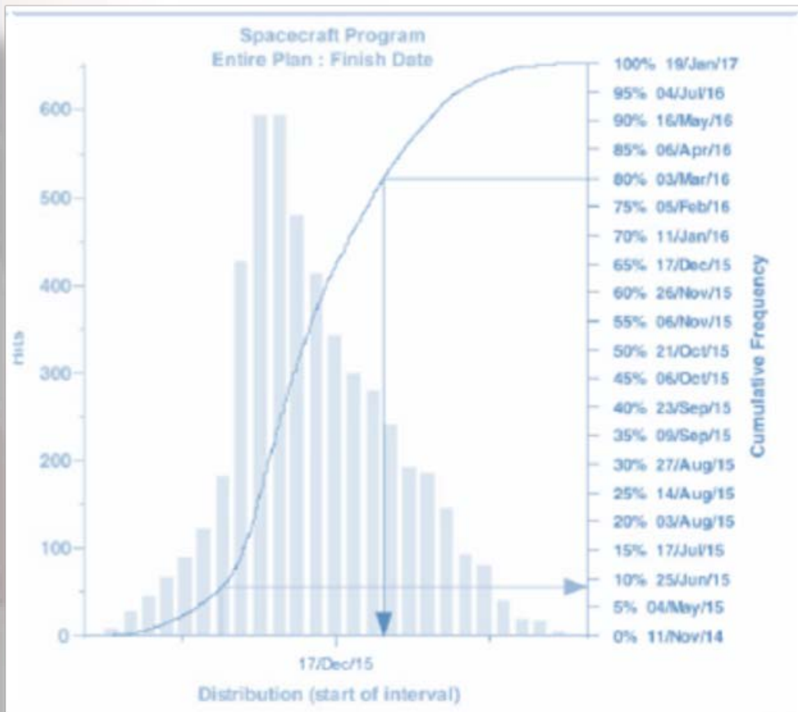


Best Practice 7: Ensuring Reasonable Total Float

- Float will be monitored only to the Release level of the schedule



Best Practice 8: Conducting a Schedule Risk Analysis



- An SRA can be done at the Release level in the schedule and considers whether extra sprints or additional teams may be needed to deliver must have features
  - Threats and opportunities should be considered including team size, the amount of distractions, availability of tools, and dependencies.
  - Uncertainty regarding number of sprints, releases, velocity, and new requirements are modeled
  - Customer feedback from retrospectives provide insight into risks and priorities
- Sprint planning sessions focus on potential risks that could affect delivery of must have features
- Continuous integration used to mitigate threats

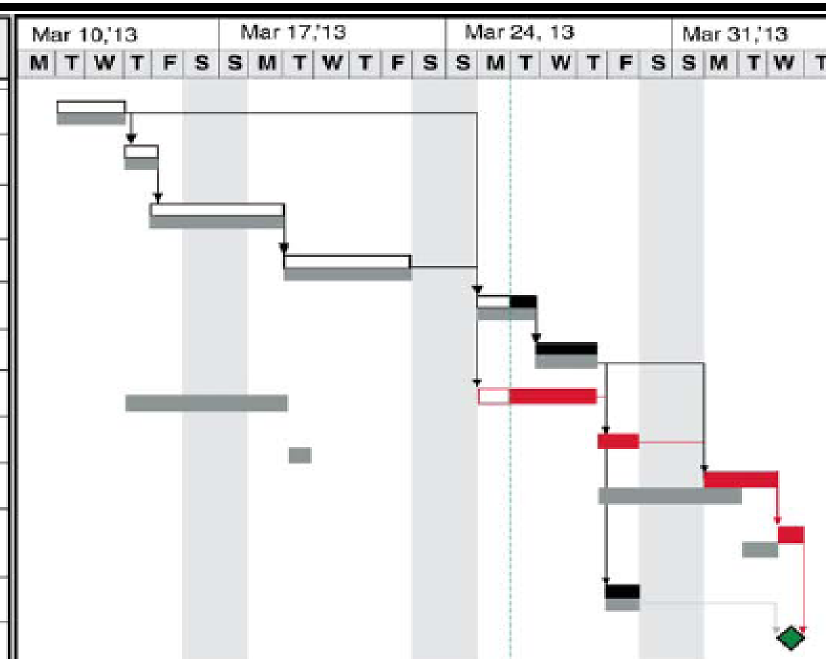
Best Practice 9: Updating the Schedule Using Actual Progress and Logic

- This best practice is maintained at the Release level
- Release progress is updated at the end of each sprint
  - Burndown / burn-up charts are quantifiable back up data for schedule progress
  - Velocity is tracked to measure the number of story points delivered per sprint
- Daily stand-up meetings are conducted to check status
- The team constantly pulls work from the product backlog to ensure that no one sits idle
- Progress board is updated at the end of each day

Best Practice 10: Maintaining a Baseline Schedule

- Roadmap and release plans become the baseline from which to measure schedule variances
- Demonstration of working software determines stakeholder / customer satisfaction
- Retrospectives are conducted to capture lessons learned and motivate development staff

Activity name	Start	Actual start	Total float (in days)
Install first floor joists and decking	3-12-13	3-12-13	0
Install waterproofing on exterior basement walls	3-14-13	3-14-13	0
Finish excavate and pour basement floor	3-15-13	3-15-13	0
Frame first floor walls	3-19-13	3-19-13	0
Install roof trusses	3-25-13	3-25-13	1
Install roof decking	3-27-13	NA	1
Exterior backfill basement walls	3-25-13	3-25-13	0
Rough grade property	3-29-13	NA	0
Form and pour driveway	4-1-13	NA	0
Finish excavate and pour garage floor	4-3-13	NA	0
Rough-in framing inspection	3-29-13	NA	3
Framing complete	4-3-13	NA	0



GAO

You’re Invited…

- GAO invites interested parties to meet with us and other experts to discuss further updates to the Guide so that it continually reflects best practices
    - If interested, please e-mail your contact info to:
        - Karen Richey - richeyk@gao.gov

Back-Up

GAO

Shared Values of Traditional and Agile Schedules

- **Traditional:** Processes and Tools
  - ***Agile:*** *Individuals and interactions*
- **Traditional:** Comprehensive Documentation
  - ***Agile:*** *Working Software/Solutions*
- **Traditional:** Contract Negotiations
  - ***Agile:*** *Customer Collaboration*
- **Traditional:** Following a plan
  - ***Agile:*** *Responding to Change*

Agile Background- Definition of Selected Terms

Term	Definition	Also called
Project Roadmap	High level view of the features the project will set out to accomplish along with the expected business value	Project Vision
Release Plan	Schedule for developing working software that identifies the expected number of sprints and features that will be included in a release	
Epics	High level capabilities	High level Requirements
Features	Next level below an epic which represents items of specific business value. Some features may need several stories to be complete.	Capabilities
User Story	Small chunk of software that identifies business value and success criteria that can be completed within a sprint timebox. A user story defines the work to be done to satisfy a feature.	
Story points	Assessed value of effort for an epic, feature, or user story based on team consensus	
Sprints	short-term, timeboxed effort for delivering an agreed upon number of story points	Iterations, increments
Product Backlog	List of prioritized user stories identified as Must Haves, Should Haves, Could Haves and Nice to Haves	Requirements backlog, feature list
Burn down Chart	Burn down charts represent completed user stories and reflect the rate of progress over time. Can be compared to estimated number of stories to be completed during each sprint for variance analysis.	Burn up chart
Retrospective	Final review of what was accomplished during a sprint and documentation of lessons learned (Agile team and customers/stakeholders attend)	
Velocity	The rate of progress accomplished by the team during a sprint (measures number of story points delivered per sprint to better estimate future work). Velocity reflects a team's cadence and will vary among teams.	Cadence

Headquarters U.S. Air Force

Integrity - Service - Excellence

USAF Data Collection and Management for MAIS Programs

11 August 2015

U.S. AIR FORCE

Richard Mabe
Air Force Cost Analysis Agency
IT and Electronics Division

U.S. AIR FORCE

Overview

- Purpose
- Background
- AIS Programs (Mil-Std-881C, App K)
- Software (SRDR)
- Way Ahead

U.S. AIR FORCE

Purpose

- Present the AF approach to improve the collection and management of MAIS program data to be used for cost estimating and analysis
- Discuss the use of commodity specific reporting formats (e.g., MAIS specific CSDR and SRDR)
- Solicit feedback from the IPT on issues being faced by the AF

U.S. AIR FORCE

Background

- In 2013, AF used Gap Analysis to identify data shortfalls in our cost databases
- Initiated effort called "Cost Estimating Modeling" to fill the gaps and support development of CERs/ methodologies
- Used an OSD CAPE contract vehicle; influencing future data collection approaches being developed by the CAPE
  - Commodity specific data collection formats for CSDR reports
  - Technical Factors (1921-T) to supplement Cost Factors (1921)
  - Tables to support revised CARD format/policies
  - MAIS specific approaches for HW and SW reporting
  - Detailed format for common items; e.g., SEPN, Data, Test, Spares, Site Activation, OGCs

U.S. AIR FORCE

AIS Specific Formats and Data (Includes SW Formats)

U.S. AIR FORCE

Overview

- Based on Mil-Std-881C, Appendix K
- Gap Analysis identified shortfalls in data related to Infrastructure and Hardware:
  - Development Environment
  - Operating Environment
  - Data Centers
  - C2 Centers/Systems
- Impact:
  - Difficult to estimate data center consolidation costs
  - No means to validate/verify DISA fee-for-service costs
  - Unable to evaluate the cost of commercial cloud capability
  - Limited ability to estimate C2 system costs (distributed systems)
- Developed templates to drive data collection
  - Coordinated content with SW and O&S data template teams
  - Template examples attached

U.S. AIR FORCE

AIS Commodity Template (Excel Workbook)

Metadata Table	Program identification
Milestone Table	Planned/Actual program milestone dates
PMP HW Technical Table	Mil-Std-881C WBS and associated Tech Factors for the Prime Mission System
HW Architecture Drivers Table	System requirements summary to use for sizing HW
Non-PMP Technical Table	SEPM, Data, Test, Site Activation, Spares, etc …
SW Development Table	SRDR Data; Use for ALL SW end items indentured to the Prime Mission System
SW Maintenance Table	Maintenance SRDR data; use for ALL SW end items indentured to the Prime Mission System
O&S Table	CAPE O&S WBS tailored to specific commodity
Other Templates (OGC, Phased Quantities, Parts, Roles, Configuration, Manpower, etc.)	Other program data

U.S. AIR FORCE

Walk-Through of Templates*

UNCLASSIFIED, FOR OFFICIAL USE ONLY

CEM Template

Milestone Table

Enter parameter values as a single date value in this column.

Identify if the parameter value is an actual value or an estimated value.

For additional columns to express alternative designs, hit Ungroup button above.

Parameter Name	Value	Units	Estimate (Planned) or Actual	Source	Notes	Definition of Parameter
Material Development Decision		DDMMMYY				Milestone Date
Completion of AoA, BCA, or Equivalent		DDMMMYY				Milestone Date
Milestone A		DDMMMYY				Milestone Date
Technology Maturation and Risk Reduction Contract Award		DDMMMYY				Milestone Date
Capability Development Document (CDD) Validation		DDMMMYY				Milestone Date
Development RFP Release Decision		DDMMMYY				Milestone Date
Risk Reduction Build		DDMMMYY				Milestone Date
Preliminary Design Review (PDR)		DDMMMYY				Milestone Date
Milestone B		DDMMMYY				Milestone Date
EMD Contract Award		DDMMMYY				Milestone Date
Increment 1..n Begin		DDMMMYY				Milestone Date
Build 1..n Begin		DDMMMYY				Milestone Date
Increment 1..n End		DDMMMYY				Milestone Date
Build 1..n End		DDMMMYY				Milestone Date
System Requirements Review (SRR)		DDMMMYY				Milestone Date
Critical Design Review (CDR)		DDMMMYY				Milestone Date
First Delivery		DDMMMYY				Milestone Date
Development Test & Evaluation Begin		DDMMMYY				Milestone Date
Development Test & Evaluation End		DDMMMYY				Milestone Date
Operational Test & Evaluation Begin		DDMMMYY				Milestone Date
Operational Test & Evaluation End		DDMMMYY				Milestone Date
Limited Deployment Decision (1..n)		DDMMMYY				Milestone Date
Integration Build 1..n		DDMMMYY				Milestone Date
Milestone C		DDMMMYY				Milestone Date
Limited Deployment (1..n)		DDMMMYY				Milestone Date
Operational Test & Evaluation Begin		DDMMMYY				Milestone Date
Operational Test & Evaluation End		DDMMMYY				Milestone Date
Full Deployment Decision (FDD)		DDMMMYY				Milestone Date
Initial Operational Capability (IOC)		DDMMMYY				Milestone Date
Full Deployment (FD)		DDMMMYY				Milestone Date
Full Operational Capability (FOC)		DDMMMYY				Milestone Date

When any parameter calls for additional detail, insert rows underneath and indent sub-names as needed.

*Embedded Excel File of Complete Workbook. Snap-shot views in back-Ups

Way Ahead

U.S. AIR FORCE

Identify MAIS Specific Data and Factors

- Identify and account for differences by Program Type; for example:
  - ERP Life Cycle:
    - Change Mgmt
    - Blue Printing
    - SW Configuration
    - Cut-Over
    - Go Live
    - Post Support
  - Other Business and Info Systems:
    - Smaller programs
    - Specific functions
  - Mission Operations Systems (C2, Intell, Cybersecurity):
    - Self Contained (not centrally hosted)
    - Unique SW Development
    - Military Operators

U.S. AIR FORCE

Identify MAIS Specific Data and Factors

- Identify and account for differences in SW development/ procurement/ sustainment
  - **Sizing**:
    - How is Effort measured? - SLOC; Function Points; Use Cases; Story Points; RICE-FW; Other
    - Current SRDR is very SLOC centric
    - Appendix K WBS is ERP/ Business System Centric
  - **Productivity**:
    - What drives Effort Hours? – SLOC; Requirements; Stories, Use Cases; Functions; Sprints
  - **Delivered Effort**:
    - What is the Delivered Product? - SW End Item; SW Release; SW Configuration Item; Other
  - **Hosting/Access**:
    - What is the best approach? – Gov't Data Center/ DODIN; Federated Servers/Processors; Cloud Access; Other

U.S. AIR FORCE

Hope to Gain from IPT

- Details from the 1-on-1 discussions to help identify MAIS specifics
- For example: Tailor WBS for MAIS Program reporting

	AIS 881C, Appendix K Expanded
WBS Number	WBS Element Name
1.0	Automated Information System (AIS) Investment
1.1	AIS Prime Mission Product Release
1.1.1	Custom Application Software
1.1.1.1	Subsystem Hardware
1.1.1.2	Subsystem Software CSCI 1…n (Specify)
1.1.1.3	Subsystem Software Integraton, Assembly, Test and Checkout
1.1.2	Enterprise Service Element
1.1.2.1	Enterprise Service Element Hardware
1.1.2.2	Enterprise Service Element Software CSCI 1…n (Specify)
1.1.2.3	Enterprise Service Element Software Integraton, Assembly, Test and Checkout
1.1.3	Enterprise Information System
1.1.3.1	Business Area Hardware
1.1.3.2	Business Area Software CSCI 1…n (Specify)
1.1.3.3	Business Area Integration, Assembly, Test and Checkout
1.1.4	External System Interface Development 1…n (Specify)
1.1.4.1	External System Interface Hardware
1.1.4.2	External System Interface Software CSCI 1…n (Specify)
1.1.4.3	External System Interface Integration, Assembly, Test and Checkout

1. Map to Domains, and expand
   - ERP Systems
   - Other Business Systems
   - Mission Operations/C2 Systems
   - Infrastructure
2. Describe HW
3. Describe SW
4. Expand on Integration/Hosting
5. Do we need to separate the WBS for the various domains and applications; e.g. – ERP specific? C2 specific?

U.S. AIR FORCE

Hope to Gain from IPT

- SRDR Considerations:
  - Other than SLOC Measures – need to expand, define and describe how to collect data for the SRDR

Non-SLOC Based Software Size
Function Point Measure
RICE-FW Object Measure
Story Points Measure
Other Measures

  - How do you track, forecast and estimate these measures?
  - Do the sizing (leading to effort) and productivity (leading to schedule) vary by vendor/developer?
    - Can they be standardized?
    - Should they be?
  - We are especially interested in your approach for defining, measuring and estimating size and productivity for Agile projects
    - What is the delivered effort? By requirements or release or sprints?
    - What are the standard practices for Agile?

Questions?

U.S. AIR FORCE

Back-Ups

U.S. AIR FORCE

Hope to Gain from IPT

- For comparison: the SLOC based portion of the SRDR is very detailed:
  - New code
  - Deleted code
  - Reuse/Carry Over with and without modification
    - % Design Modifications
    - % Code and Unit Test (CUT) Modifications
    - % Integration Modifications
    - Adaptation Adjustment Factor (AAF)
- How do we expand the detail for the non-SLOC measures to be of the same quality/completeness?
  - What data is available in your organizations?
  - What are you willing to report?
  - What advice do you have for government cost estimators?

U.S. AIR FORCE

Milestone Table

UNCLASSIFIED, FOR OFFICIAL USE ONLY

CEM Template

Milestone Table

Enter parameter values as a single date value in this column.

Identify if the parameter value is an actual value or an estimated value.

For additional columns to express alternative designs, hit Ungroup button above.

Parameter Name	Value	Units	Estimate (Planned) or Actual	Source	Notes	Definition of Parameter
Material Development Decision		DDMMMYY				Milestone Date
Completion of AoA, BCA, or Equivalent		DDMMMYY				Milestone Date
Milestone A		DDMMMYY				Milestone Date
Technology Maturation and Risk Reduction Contract Award		DDMMMYY				Milestone Date
Capability Development Document (CDD) Validation		DDMMMYY				Milestone Date
Development RFP Release Decision		DDMMMYY				Milestone Date
Risk Reduction Build		DDMMMYY				Milestone Date
Preliminary Design Review (PDR)		DDMMMYY				Milestone Date
Milestone B		DDMMMYY				Milestone Date
EMD Contract Award		DDMMMYY				Milestone Date
Increment 1..n Begin		DDMMMYY				Milestone Date
Build 1..n Begin		DDMMMYY				Milestone Date
Increment 1..n End		DDMMMYY				Milestone Date
Build 1..n End		DDMMMYY				Milestone Date
System Requirements Review (SRR)		DDMMMYY				Milestone Date
Critical Design Review (CDR)		DDMMMYY				Milestone Date
First Delivery		DDMMMYY				Milestone Date
Development Test & Evaluation Begin		DDMMMYY				Milestone Date
Development Test & Evaluation End		DDMMMYY				Milestone Date
Operational Test & Evaluation Begin		DDMMMYY				Milestone Date
Operational Test & Evaluation End		DDMMMYY				Milestone Date
Limited Deployment Decision (1..n)		DDMMMYY				Milestone Date
Integration Build 1..n		DDMMMYY				Milestone Date
Milestone C		DDMMMYY				Milestone Date
Limited Deployment (1..n)		DDMMMYY				Milestone Date
Operational Test & Evaluation Begin		DDMMMYY				Milestone Date
Operational Test & Evaluation End		DDMMMYY				Milestone Date
Full Deployment Decision (FDD)		DDMMMYY				Milestone Date
Initial Operational Capability (IOC)		DDMMMYY				Milestone Date
Full Deployment (FD)		DDMMMYY				Milestone Date
Full Operational Capability (FOC)		DDMMMYY				Milestone Date

When any parameter calls for additional detail, insert rows underneath and indent sub-names as needed.

U.S. AIR FORCE

Architecture Drivers

System Requirements	Value	Estimate or Actual	Description
Software System Type			
Database Management System (DBMS)	DBMS name		Software system that uses a standard method of cataloging, retrieving, and running queries on data
Enterprise Resource Planning (ERP)	ERP name(s)		Business process management software that allows an organization to use a system of integrated applications to manage the business and automate back office functions.
Other	Describe		
Total Numbers of Users			
Heavy / Power / Advanced	qty		Individual that operates a computer or device with advanced skills, knowledge, experience and capabilities. (http://www.techopedia.com/definition/1784/power-user)
Medium	qty		Everyday system user without advanced competencies; uses applications at a basic to moderate skill level
Light	qty		Not an everyday user of the system; uses only most basic system capabilities
Number of Concurrent Users	qty		Number of users accessing the system at the same time
Throughput (TPH)	qty		Peak Number of transactions per hour
Response time (seconds)	time (sec)		Expected time between sending a request (transaction) and receiving a response
Think time (seconds)	time (sec)		average time user spends between transactions
Size of request transaction	size (Mb)		Average size of a request sent to the server
Size of response transaction	size (Mb)		Average size of a response sent from the server to the user
Are transactions saved / stored? (Y/N)	Y/N		
How long are transactions stored?	time		Specify length of time transactions must remain in storage (days, weeks, months, years, etc)
System Availability (0-100%)	(0-100%)		A Software system possesses 100% availability if it can service a request within the defined response time for every user.
IT Security level Requirements (FIPS 1-4)	FIPS 1-4		See Template Definitions
System Complexity			
Low / Simple			**Simple Complexity:** Low complexity application. Well-documented; Easily configured interfaces; easily integrated into workflows
Medium			**Medium Complexity:** ; Average complexity application; Adequately documented; Moderate effort to configured interfaces; May expose some service based interfaces; minimal glue code required
High			**High Complexity:** Complex application with large number of interfaces; Not well-documented. Complex to configure interfaces; Requires adapters and significant glue code to integrate
System Growth			
User Growth per Year	% or value		Expected user growth per year
Transaction Growth per Year	% or value		Expected growth of transactions per year
Deployment Strategy			
Centralized Deployment			All main server hardware is hosted / housed in a single environment
Decentralized Deployment			Server nodes are distributed at individual sites and data is uploaded regularly to a master server location
HW System Configuration (Tier 1, 2, or 3)			
Tier 1			System servers perform all system services
Tier 2			System servers perform designated system services (e.g. servers dedicated to web services and servers dedicated to application and data management services)
Tier 3			System servers are further differentiated into three categorical functions (e.g. web servers, applications servers, and data management servers)
Network Speed (T1, DSL, Modem, etc...)			What is the maximum network speed for processing user transactions (e.g. 3 T1 lines, etc.)
Number/name of system sites/environments			The name or qty of system sites / environments where AIS HW is hosted (e.g. Production Environment, COOP, etc...)
Hardware constraints			List any hardware constraints due to program or software requirements.

U.S. AIR FORCE

PMP HW Table

WBS Number	WBS Element	Parameter name	Value	Units	Estimated or Actual	Source	Notes
1.1.5	Development Suite Hardware (HW)						
1.1.5.1	Servers						
1.1.5.1.1	Web Servers						
		Vendor/Original Equipment		Text			
		System / Type		Text			
		Site/Environment		Text			
		Buy Year		YYYY			
		Total QTY this project		Units			
		Total QTY in common buy					
		Processor		Text			
		# Cores					
		# Chips					
		# Cores/Chip					
		# Threads/Core					
		Megahertz (MHz)		MHz			
		Ram (GB)		Gigabytes (GB)			
		Operating System		Text			
		Warranty (in months)		Months			
		Warranty (Scope)		Text			
1.1.5.1.2	Application Servers						
		Vendor/OEM		Text			
		System / Type		Text			
		Site/Environment		Text			
		Buy Year		YYYY			
		Total QTY this project		Units			
		Total QTY in common buy					
		Processor		Text			
		# Cores					
		# Chips					
		# Cores/Chip					
		# Threads/Core					
		MHz					
		RAM (GB)					
		Operating System					
		Warranty (in months)					
		Warranty (Scope)					
1.1.5.1.3	Database Servers						
		Vendor/OEM					

Enter parameter values as a single value in this column.

Where did the data come from (document, contractor,

Notes should include the following:
- Any information to help understand any differences from the source data and the definition.
- Analogous system if used.

Also Includes:
Other Servers
Data Storage
Work Stations
Comm HW/LANs
Crypto Devices
Peripherals
System Integration

U.S. AIR FORCE

Non-PMP Table

WBS Number	WBS Element	Parameter Name	Value	Estimated or Actual	Source	Notes
1.2	System Engineering					
		Number of Interfaces which will be integra				
		Number of Systems Engineering related C				
		Number of "shalls" in requirements docum				
		Document from which "shalls" were count				
		Provide Quantitative Staffing Data				
		Staffing Level				
		Peak or Average				
		Systems Engineering Start Date				
		Systems Engineering End Date				
		Amplifying Descriptive Text				
		Total Systems Engineering Hours				
1.3	Program Management					
		Number of Major Subcontractors/Suppliers				
		Number of Customer Stakeholders				
		Number of Program Management related CDRLs				
		Provide Quantitative Staffing Data				
		Staffing Level				
		Peak or Average				
		Program Management Start Date				
		Program Management End Date				
		Amplifying Descriptive Text				
		Total Program Management Hours				
1.4	Change Management	Provide Quantitative Staffing Data				
		Staffing Level				
		Peak or Average				
		Program Management Start Date				
		Program Management End Date				
		Amplifying Descriptive Text				
		Total Program Management Hours				
1.5	System Test and Evaluation					

Enter parameter values as a single value in this column.

Where did the data come from (document, contractor, Government, etc.)

Notes should include the following:
- Any information to help understand any differences from the source data and the definition.
- Analogous system if used.
- If another estimate exists - value and source for other estimate i.e., Software SLOC - Program office value in the table and contractors value in the notes or vice versa.

Also Includes:
- Training
- Data
- PSE
- CSE
- Ops/Site Activation
- Site Conversion
- Spares

U.S. AIR FORCE

SW Development (SRDR Data)

WBS/CRS Element	WBS/CRS Name		CSCI 1				
Parameter Name	Unit of Measure	Value	Value	Low	High	Margin	Estimated or Actual
Report Context Release Level							
Release Begin Date							
Release End Date							
Software Release #							
Software Process Maturity							
Internal Software Requirements Count Definition							
External Interface Requirements Count Definition							
Product Quality Reporting Definition							
Precedents							
Precedent 1							
Precedent description 1							
Precedent 2							
Precedent description 2							
Precedent 3							
Precedent description 3							
Hours per Staff Month	Hours						
Hours per Staff Month Type							
Contractor Defined SW Development Activities							
SW Dev Activity 1							
ISO 12207:2008 Activity 1							
ISO 12207:2008 Activity 2							
ISO 12207:2008 Activity 3							
SW Dev Activity 2							
ISO 12207:2008 Activity 1							
ISO 12207:2008 Activity 2							
ISO 12207:2008 Activity 3							
Code Counter Version							
Alternative Code Counter							
Report Context CSCI Level							
CSCI Start Date							
CSCI End Date							
Primary Development Organization							
Outsourced Development Organization							
Name 1							

Enter Release parameter values as a single value in this column.

Enter CSCI parameter values as a single value in these columns.

Identify i parameter va actual valu estimated

Also Includes:
Product Description
Product Size
- **Internal Req'ts**
- **External Req'ts**
- **Code (SLOC)**
- **Non-SLOC**

Resource& Schedule
Product Quality

UNCLASSIFIED

U.S. AIR FORCE

CSDR Plans Comparison
Page 1 DD Form 2794

10. WBS ELEMENT CODE a. PROGRAM	b. CONTRACT	11. WBS REPORTING ELEMENTS	12a. CONTRACTOR NAME	12b. CONTRACT NUMBER	13. REPORTS REQUIRED (X if applicable) a. CWBS DICTIONARY	b. DD 1921 (CDSR)	c. DD 1921-1 (FCHR)	d. DD 1921-2 (PCR)	e. SRDR FORMATS
1.0	1.0	Small Diameter Bomb Increment II	Raytheon Missile Systems Tucson, AZ	FA8672-10-C-0002	X	X	X		X
1.1	1.1	Complete Round			X	X			X
1.1.1	1.1.1	Structure (Over Body Assembly)			X	X	X		
	1.1.1.1	Wings			X	X			
	1.1.1.2	Wing Deployment Driver			X	X			
	1.1.1.3	Air Turbine Alternator			X	X			
	1.1.1.4	Over Bomb Harness			X	X			
	1.1.1.5	Over Body Structure			X	X			
	1.1.1.6	Structure IAT&CO			X	X			
1.1.2	1.1.2	Payload			X	X	X		
	1.1.2.1	Warhead			X	X			
	1.1.2.2	Fuze			X	X			
	1.1.2.3	Payload IA&T			X	X			
1.1.3	1.1.3	Guidance and Control							
	1.1.3.1	Seeker Section Subsystem							
	1.1.3.1.1	Multi-Mode Seekerhead (MMW/IR/SAL)							
	1.1.3.1.2	GPS Receiver/Navigator (GRN)							
	1.1.3.1.2.1	GRN (excluding software)							
	1.1.3.1.2.2	GRN Software							
	1.1.3.1.3	Forward Electronics							
	1.1.3.1.4	Forward Electronics Structure							
	1.1.3.1.5	Forward Assembly Harnesses							
	1.1.3.1.6	GPS Antennas							
	1.1.3.1.7	Seeker Algorithms							
	1.1.3.1.8	Seeker Section IAT&CO							
	1.1.3.2	Control Devices (Aft Section Subsystem)							
	1.1.3.2.1	Control Actuation System							
	1.1.3.2.2	Data Link							
	1.1.3.2.2.1	Data Link (excluding software)							
	1.1.3.2.2.2	Data Link Software							
	1.1.3.2.3	Aft Electronics							
	1.1.3.2.4	Battery							
	1.1.3.2.5	Aft Structure							
	1.1.3.2.6	UHF Antenna							
	1.1.3.2.7	Aft Section Harnesses							
	1.1.3.2.8	Guidance and Control Algorithms							
	1.1.3.2.9	Aft Section IAT&CO							
	1.1.3.3	Guidance and Control Application Software							
	1.1.3.3.1	Seeker Section Operation Flight Program (Skr OFP							
	1.1.3.3.2	Aft Section OFP							
	1.1.3.3.3	OFP Integration							
	1.1.3.3.4	OFP Test and Support							
1.1.4	1.1.4	Fuze - N/A (included under Payload)							
1.1.5	1.1.5	Safety/Arm - N/A							
1.1.6	1.1.6	Propulsion - N/A							

SDB Program example.
No reporting at CSCI level, SRDR does *not* tie to 1921

As-Is

11. WBS ELEMENT CODE a. PROGRAM/ CONTRACT/ SUBCONTRACT	b. CONTRACT/ SUBCONTRACT	12. WBS REPORTING ELEMENTS	13. REPORTS REQUIRED (X if applicable) DD 1921-3 (CBDR): ☐ a. CWBS DICTIONARY	b. DD 1921 (CDSR) To Date	b. DD 1921 (CDSR) Estimate At Completion	c. DD 1921-1 (FCHR)	d. DD 1921-2 (PCR)	e. DD 1921-5 (SFCHR)	f. SRDR FORMATS
1.0	1.0	Missile System	X	X	X	X			
1.1	1.1	Air Vehicle	X	X	X	X	X		
1.1.1	1.1.1	Airframe	X	X	X	X	X		
1.1.1.1	1.1.1.1	Airframe Integration, Assembly, Test and Checkout	X	X	X				
1.1.1.2	1.1.1.2	Primary Structure	X	X	X				
1.1.1.3	1.1.1.3	Secondary Structure	X	X	X				
1.1.1.4	1.1.1.4	Aero-Structures	X	X	X				
1.1.1.5	1.1.1.5	Other Airframe Components 1…n (Specify)	X	X	X				
1.1.2	1.1.2	Propulsion	X	X	X	X	X		
1.1.2.1	1.1.2.1	Propulsion Integration, Assembly, Test and Checkout	X	X	X				
1.1.2.2	1.1.2.2	Motor/Engine	X	X	X				
1.1.2.3	1.1.2.3	Thrust Vector Actuation	X	X	X				
1.1.2.4	1.1.2.4	Attitude Control System	X	X	X				
1.1.2.5	1.1.2.5	Fuel/Oxidizer Liquid Management	X	X	X				
1.1.2.7	1.1.2.7	Flight Termination/Mission Termination	X	X	X				
1.1.2.6	1.1.2.6	Arm/Fire Device	X	X	X				
1.1.2.8	1.1.2.8	Propulsion Software Release 1…n	X	X	X	X			X
1.1.2.8.1	1.1.2.8.1	Propulsion Software Release 1…n CSCI 1…n	X	X		X			X
1.1.2.9	1.1.2.9	Other Propulsion Subsystems 1…n (Specify)	X	X	X				
1.1.3	1.1.3	Power and Distribution	X	X	X	X	X		
1.1.3.1	1.1.3.1	Power and Distribution Integration, Assembly, Test and Checkout	X	X	X				
1.1.3.2	1.1.3.2	Primary Power	X	X	X				
1.1.3.3	1.1.3.3	Power Conditioning Electronics	X	X	X				
1.1.3.4	1.1.3.4	Distribution Harness	X	X	X				
1.1.3.5	1.1.3.5	Power and Distribution Software Release 1…n	X	X	X	X			X
1.1.3.5.1	1.1.3.5.1	Power and Distribution Software Release 1…n CSCI 1…n	X	X		X			X
1.1.3.6	1.1.3.6	Other Power and Distribution Subsystems 1…n (Specify)	X	X	X				
1.1.4	1.1.4	Guidance	X	X	X	X	X		
1.1.4.1	1.1.4.1	Guidance Integration, Assembly, Test and Checkout	X	X	X				
1.1.4.2	1.1.4.2	Dome Assembly	X	X	X				
1.1.4.3	1.1.4.3	Seeker Assemblies (IR Seeker)	X	X	X				
1.1.4.3.1	1.1.4.3.1	Focal Plane Array	X	X					
	1.1.4.3.2	Optics	X	X					
	1.1.4.3.3	Cooling	X	X					
1.1.4.3.4	1.1.4.3.4	Gimbal	X	X					
1.1.4.3.5	1.1.4.3.5	Seeker Electronics (Specify)	X	X					
1.1.4.3.6	1.1.4.3.6	Seeker Structure (Specify)	X	X					
1.1.4.3.7	1.1.4.3.7	Seeker Integration, Assembly, Test and Checkout	X	X					
1.1.4.3	1.1.4.3	Seeker Assemblies (RF Seeker)	X	X	X				
1.1.4.3.1	1.1.4.3.1	Antenna	X	X					
1.1.4.3.2	1.1.4.3.2	Transmitter	X	X					
1.1.4.3.3	1.1.4.3.3	Receiver/Exciter	X	X					
1.1.4.3.4	1.1.4.3.4	Gimbal	X	X					
1.1.4.3.5	1.1.4.3.5	Seeker Electronics (Specify)	X	X					
1.1.4.3.6	1.1.4.3.6	Seeker Structure (Specify)	X	X					
1.1.4.3.7	1.1.4.3.7	Seeker Integration, Assembly, Test and Checkout	X	X					
1.1.4.3	1.1.4.3	Seeker Assemblies (Laser Seeker)	X	X	X				
1.1.4.4	1.1.4.4	Guidance Software Release 1…n	X	X	X	X			X
1.1.4.4.1	1.1.4.4.1	Guidance Software Release 1…n CSCI 1…n	X	X		X			X
1.1.4.5	1.1.4.5	Other Guidance Subsystems (Mission Computer)	X	X	X				
1.1.4.5	1.1.4.5	Other Guidance Subsystems 1…n (Specify)	X	X	X				
1.1.5	1.1.5	Navigation	X	X	X	X	X		
1.1.5.1	1.1.5.1	Navigation Integration, Assembly, Test and Checkout	X	X	X				
1.1.5.2	1.1.5.2	Sensor Assemblies	X	X	X				
1.1.5.3	1.1.5.3	Navigation Software Release 1…n (Specify)	X	X	X	X			X
1.1.5.3.1	1.1.5.3.1	Navigation Software Release 1…n (Specify) CSCI 1…n	X	X		X			X
1.1.5.4	1.1.5.4	Other Navigation Subsystems 1…n (Specify)	X	X	X				

To-Be

Missile Template.
Reporting at CSCI level, SRDR ties to 1921

U.S. AIR FORCE

CSDR Plans Comparison Page 2 DD Form 2794

14. CSDR SUBMISSION DATES

a. SUBMISSION	b. FORM(S)	c. EVENT	d. AS OF DATE (YYYYMMDD)	e. DUE DATE (YYYYMMDD)
1	CWBS Dictionary	60 days after Contract Award		
2	CWBS Dictionary (Revision)	In parallel with any other submission event when CWBS has changed		
3	1921 and 1921-1 (Initial)	60 days after Contract Award; Or 60 days after start of each Software		
4	1921 and 1921-1 (Event Submission)	At Prelimiary Design Review		
5	1921 and 1921-1 (Event Submission)	At Critical Design Review		
6	1921 and 1921-1 (Annual)	Annual Report - Year 1…n (reset 12-month cycle with each event		
7	1921 and 1921-1 and 1921-2 (Final)	Contract Completion		
8	SRDR Development (Initial)	60 days after Contract Award; Or 60 days after start of each Software		
9	SRDR Development (Event Submission)	Sizing-only (Requirements and SLOC) updates at SRR, PDR, and CDR		
10	SRDR Development (Annual)	Annual After CDR - Year 1…n		
11	SRDR Development (Final)	Final report at the end of each SW release		
12	SRDR Development (Final)	Contract Completion - Completion of all SW releases		
13	SRDR Maintenance (Final)	Annual Report - Year 1…n (all Releases finished in a given year)		
14	1921-T (Event Submission)	At Preliminary Design Review		
15	1921-T (Event Submission)	At Critical Design Review		
16	1921-T (Event Submission)	30 days after First Production Representative Unit		

Same reporting events for 1921 and SRDR

Need to be in sync with 1921-5?

15. REMARKS

The contractor will utilize the specific definitions to segregate recurring and non-recurring costs based on Attachment A to DD Form 2794 instructions.
The contractor will refer to specific instructions in the Software Resources Data Reporting (SRDR) Software Development (DI-MGMT-XXXX) and Software Maintenance DIDs (DI-MGMT-XXXX) to separately report Software specific SE/PM and other common elements in the WBS/CRS.
The contractor will submit CCDRs and SRDRs simultaneously in every specified submission event stated in DD Form 2794.
Provide attachment showing unit cost for each item $1000 or greater, and for any other required for variant visibility.

Report SW-Specific Common Elements

U.S. AIR FORCE

AFCAA CEM Software

CSDR Mapping

WBS	CCDR	SRDR
1.0 Project	X	
1.1 PMP	X	
1.1.1 HW	X	
1.1.2 SW Release 1..n	X	X
1.1.2.1 SW CSCI 1..n	X	X
1.2 SEPM	X	
1.2.1 HW SEPM	X	
1.2.2 SW SEPM	X	X
1.3 IAT&C	X	
1.3.1 HW IAT&C	X	
1.3.2 SW IAT&C	X	X
1.4 ST&E	X	
1.4.1 HW ST&E	X	
1.4.2 SW ST&E	X	X
1.5 Other Common Elements (OCE)	X	
1.5.1 HW OCE	X	
1.5.2 SW OCE	X	X
X.0 Sustainment	X	
X.5.2 Software Maintenance	X	X

SRDR will have common elements at the release level. Common Elements should tie back to 1921 reporting

U.S. AIR FORCE

O&S Table

WBS Number	WBS Element	Parameter name	Value	Units	Estimated or Actual	Source	Notes
2.1	**Unit-Level Manpower**						
2.2	**Unit Operations**						
2.3	**Maintenance**						
2.4	Sustaining Support						
2.4.1	System-Specific Training						
2.4.2	Support Equipment Replacement and Repair						
2.4.3	Sustaining/Systems Engineering						
		Number of CSCIs modified (annually					
		Tech Refresh Year (Y/N)					
		Number of SE related CDRLs					
2.4.4	Program Management						
		Number of Major Subcontractors/Suppliers					
		Number of Customer Stakeholders					
		Number of PM related CDRLs					
2.4.5	Information Systems						
2.4.5.1	Tech Refresh						
2.4.5.1.1	Servers						

Also Includes:

- Licenses
- System Support/Help Desk
- Data and Tech Pubs
- Simulator Ops

Continuing Sys Improvements

- HW
- SW
  - Custom Apps
  - Enterprise Svc
  - Business Area
  - External Interfaces

End

Agile Project Management Controls

Inherent Issues of Agile EVM and a Viable Solution for Effective Agile Project Controls

SW and IT Cost IPT Conference
Arlington, VA
August 2015

Omar Mahmoud
President
Barakah Consulting Group

Note: The data and analysis contained within this presentation is for illustrative purposes only and does not reflect actual data from any Government or Commercial project.

Discussion Topics

- Agile Intro
- Challenges Implementing EVM on Agile SW Dev. Projects
- Comparing EVM and an Application Lifecycle Management (ALM) Tool
- Summary
- Q&A

Discussion Topics

- Agile Intro
- Challenges Implementing EVM on Agile SW Dev. Projects
- Comparing EVM and an Application Lifecycle Management (ALM) Tool
- Summary
- Q&A

What is "Agile" software development?

- What is "Agile" Software Development?
  - A software development philosophy based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams
  - Promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change
- Agile Principles
  - Customer satisfaction through early and continuous delivery of valuable software
  - Welcoming changing requirements, even late in development
  - Deliver working software frequently
  - Working software is the primary measure of progress

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

**Manifesto for Agile Software Development © 2001*

Agile Concepts and Terms

- **User Stories**: A description of a user-oriented capability containing just enough information so that developers can have a reasonable understanding of how they would need to implement
- **Complexity Points**: A relative measure of complexity that enables the team to quantify the scope or effort for an issue/artifact (e.g. story)
- **Sprint / Iteration / Release**: *Sprint* - Fixed time-box in which development occurs (usually 2 - 4 weeks); *Iteration* – Collection of sprints that result in prototype software demoed/tested with users; *Release* - Multiple iterations that fulfill a major subset of requirements constituting an operational product released to the users
- **Velocity:** Performance / productivity measure that indicates progress toward capability delivery (i.e., Complexity Points completed per sprint)
- **Project / Sprint Backlog:** A prioritized database that summarizes the issues/artifacts yet to be completed for a sprint or the entire project
- **Burndown:** The concept, often shown as a graph over time, of working off or “earning” Complexity Points toward iteration or delivery completion

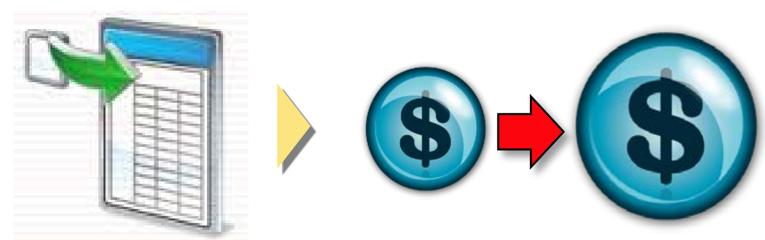
Industry trend towards agile software development methods presents challenges for PMs in managing cost, schedule, and scope

Agile Development

- Requirements flexibility
- Frequent delivery of software and interactions with product owners
- Scrum team defines processes to prioritize, estimate, and execute

PM Challenges

- Managing scope creep while maintaining flexibility
- Evaluating and reporting progress
- Projecting cost and schedule
- Prioritizing remaining work



Discussion Topics

- Agile Intro
- Challenges Implementing EVM on Agile SW Dev. Projects
- Comparing EVM and an Application Lifecycle Management (ALM) Tool
- Summary
- Q&A

ANSI-748 (EVM Compliance) focuses on meeting requirements[1] , which, in an Agile context, presents key operating challenges

Category	EVM Requirements	Agile Operating Challenge Description
Organization	‣ **Define** specific work breakdown structure (**WBS**) and **organize work into work packages that align to cost/schedule baseline**	‣ Adhering to a WBS and work organized by CAMs is too restricting to an agile development team **(*Responding to change over following a plan*)**
Planning, Scheduling, and Budgeting	‣ **Establish** and maintain a **time-phased budget baseline** and identify **management reserves** and **undistributed budget** and ensure that the contract budget base (**CBB**) is **reconciled** with the total allocated budget (**TAB**)	‣ Customer feedback and **in-progress work/resource re-prioritizations** would require **constant budget re-baselines**. EVM processes for allocating undistributed budget when needed would **limit the agility** with which an agile team can react to their changing environment. **(*Customer collaboration over contract negotiation*)**
Analysis and Management Report	‣ Provide **monthly reports** that is **reconcilable with the approved accounting system** ‣ **Provide variance reporting** of budget (PV), earned value (EV), and actuals (AC)	‣ Requires **compliant EVM software suite** (e.g., Deltek COBRA combined with SAP) and **approved process** ‣ **EV metrics** are typically **too high-level** and time-lagged to provide adequate, timely insight into agile project progress or problem causes **(*Individuals and interactions over processes and tools*)**
Revisions and Data Maintenance	‣ Incorporate **authorized changes** and record impacts **in a timely manner** ‣ Provide **reconciliation and revision reports and control and document changes**	‣ With the number of potential changes on an agile project on a sprint-by-sprint basis, a large amount of LOE is required of the performer to record, review, and receive approval for changes tied to a fluctuating WBS and schedule **(*Working software over comprehensive documentation*)**

(1) ANSI-748 has a set of 32 guidelines. For discussion purposes, we have identified operating challenges associated with an Agile development environment at a higher level in a few key areas.

EVM poses several problems for baselining and evaluating Agile software development projects

- **Estimate Product Backlog**: Counter to the way most Agile teams plan Sprints, EVM requires estimating the entire backlog
  - Changes in or additions to the backlog would necessitate a re-baseline of the PMB and re-estimation of the backlog
- **Non-Intuitive Productivity Metrics**:

EVM

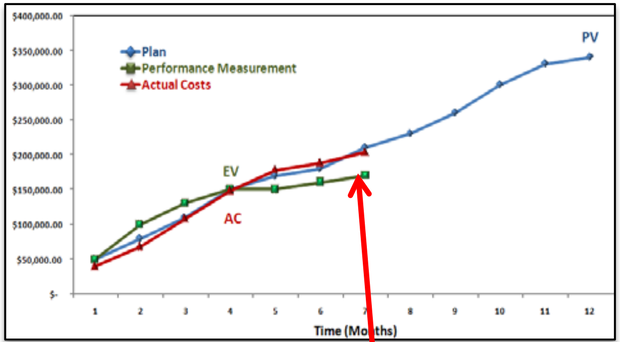
"The KTR's current SPI of .95 indicates that they are slightly behind schedule. They require a TCPI of 1.18 to complete remaining work within budget."

Versus

Agile Productivity Metric

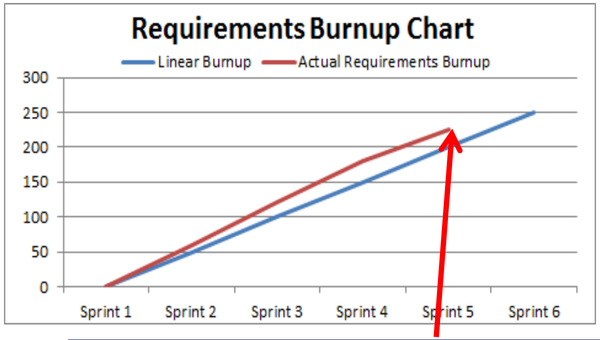
"The contractor's current velocity of 280 is 20 points below their historical average. They require a velocity of 320 to complete all req'ts within budget."

- **Inadequate insight at the Issue/Artifact level**:

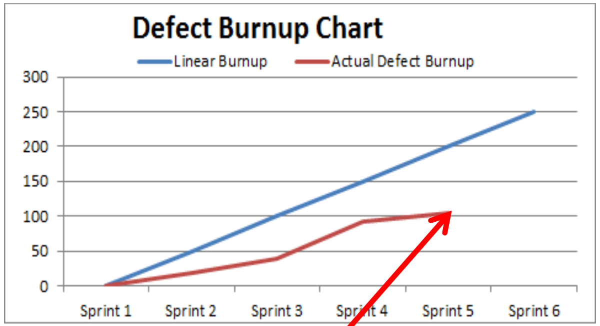


Overall project is behind

Versus



Ahead in completing Req'ts

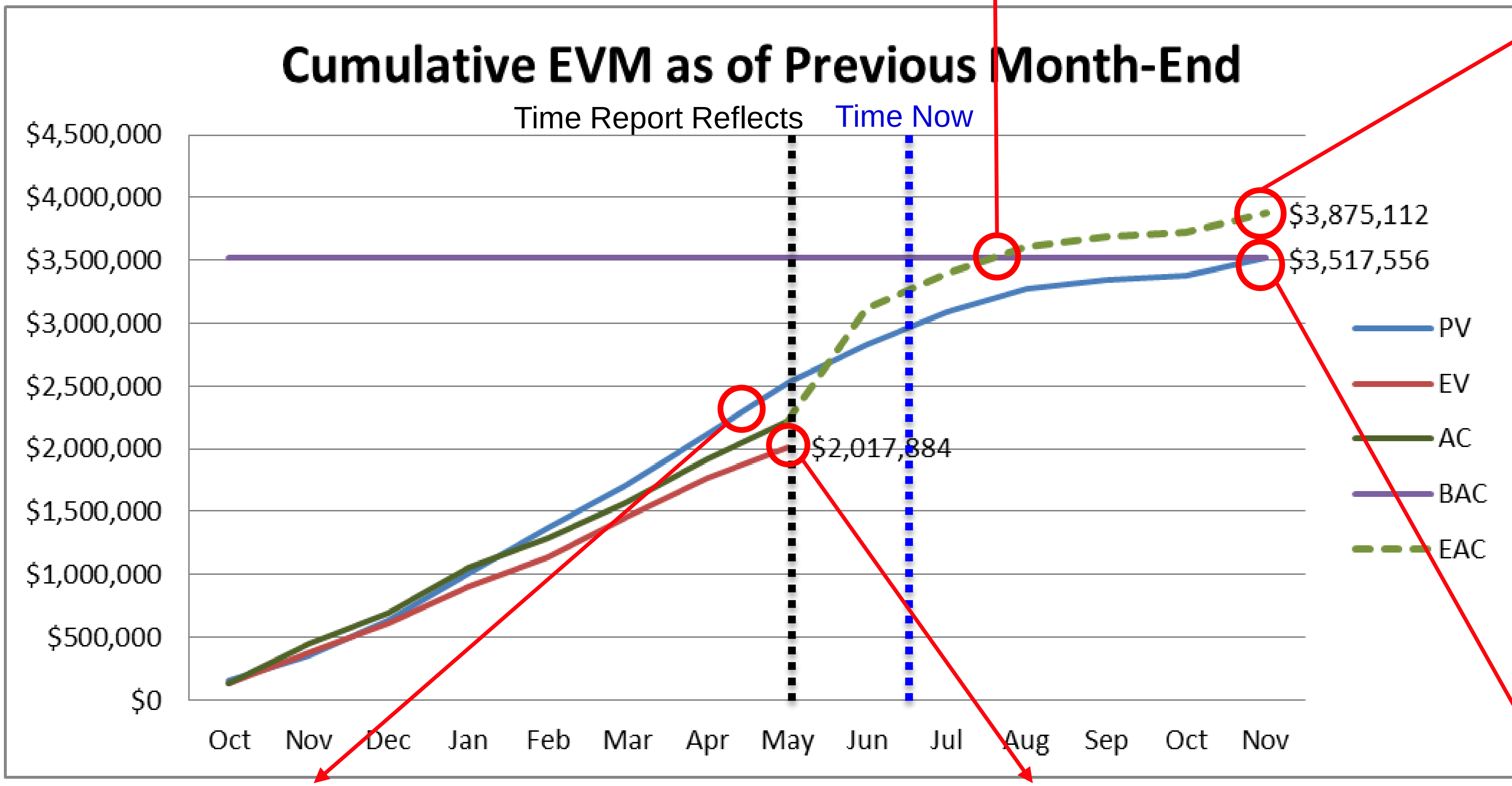


Behind in correcting Defects

EVM reports are often too-time lagged and high-level to provide timely analysis on agile projects with any insight into problem causes

What work wouldn't get done if no more funds were added?

What additional schedule (and cost) would be required if they continued at historical or any other lower burn rates?



Measure	Value
SPI	0.80
CPI	0.91
SV	-$ 506,156
SV%	-20%
CV	-$ 205,116
CV%	-10%
BAC	$ 3,517,556
EAC	$ 3,875,112
VAC	-$ 357,556

Has this plan been adjusted for the defects found during GAT of the last drop or usability findings from the recent HSI evaluation?

What's causing us to be behind? Is velocity lower than expected? Is work more complex than expected? Was harder work reprioritized earlier?

What scenarios in terms of projected cost, schedule, **and scope** could/should we plan for?

Discussion Topics

- Agile Intro
- Challenges Implementing EVM on Agile SW Dev. Projects
- Comparing EVM and an Application Lifecycle Management (ALM) Tool
- Summary
- Q&A

Use an Application Lifecycle Management (ALM) tool for capturing similar metrics as EVM with additional benefits at a lower cost

Metrics and Analysis	EVM	Agile PM Tool
ANSI-748 Compliance	x	
a) PV	x	x
b) EV	x	x
c) AC	x	x
d) Cost Curve with PV, EV, and AC	x	x
e) Metrics: CV, CPI, SV, SPI, etc.	x	x
f) Variances > 10%	x	x
g) Performance Goals?	x	x
h) Corrective Actions	x	x
IBR	x	x
Dynamic Velocity and Productivity Projections		x
Scope Trade-off Analysis		x
Buffering for Pop-ups		x
Issue Tracking and Trendlines		x

- Task Order Requirement
- Additional Value-Added Info Provided by an ALM

- ANSI-748 Compliance
  - High maintenance costs overtime
  - Implements requirements that are restrictive to an Agile environment
    - i) Long-term maintenance of a WBS
    - ii) Difficult to trace progress at Sprint level
    - iii) Constant need to re-baseline
  - Utilizing an ALM mitigates Compliance Issues
    - i) Provide funding summary report to include cost in tool with PM overhead costs for auditability
    - ii) Documentation compliance by cataloging each sprint's backlog report for change history
    - iii) Trace back to Issue ID level for each Sprint
- Data collection and Implementation of Issues using data in an ALM is less costly than EVM
  - Est. EVM LOE (Small SW Project): **1 - 2 FTE**
  - ALM Implementation LOE: **0.25 - 0.5 FTE**

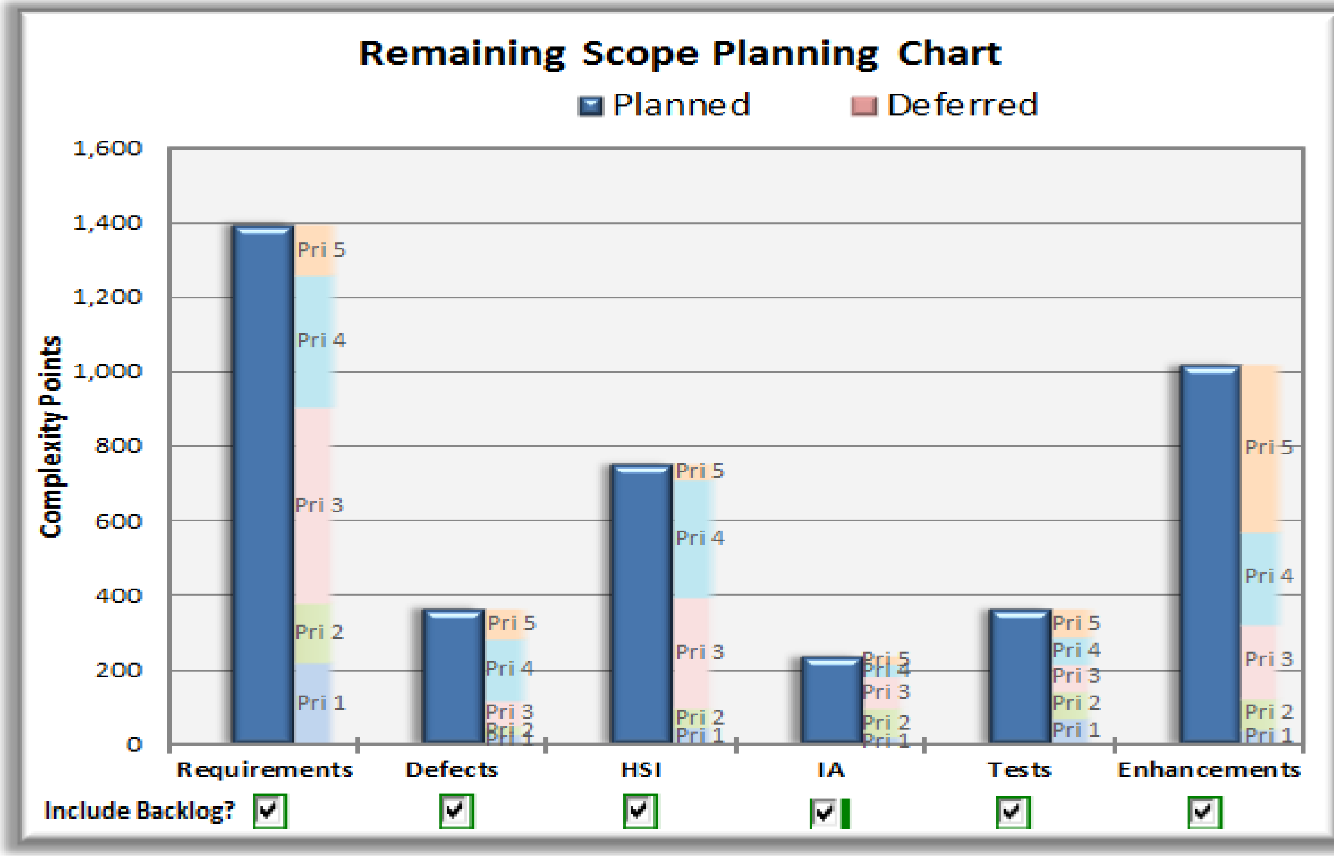
The following set of slides are an illustrative depiction of the type of analysis typically provided by an ALM tool in each sprint

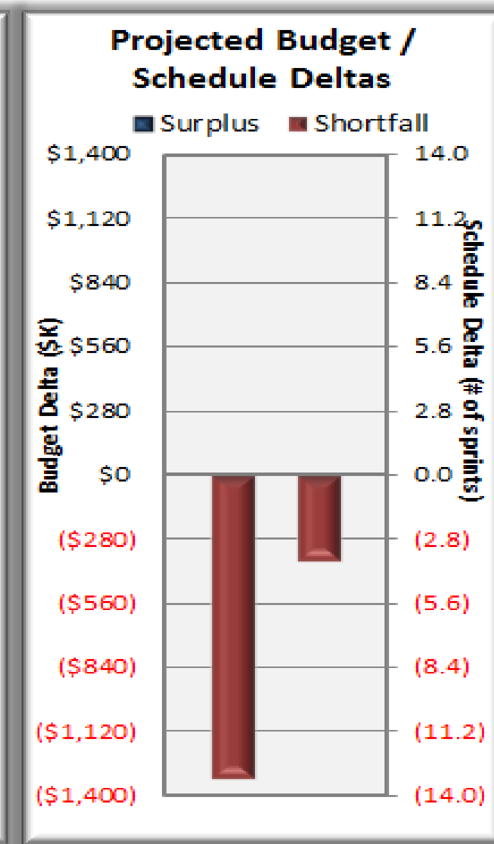
- The following slides are for illustrative purposes only and are used to provide insight into the progress the software development team is making from a budget, cost, schedule, and scope perspective, including EVM-Like metrics and more

- All of the content within this presentation is obtained from a sample set of date from an ALM tool

- Situation
  - ACME software development organization has just completed Sprint #10 our of a 20 Sprint project plan
  - Metrics and Analysis from Sprint #10 have been gathered and analyzed and projections have been completed

An ALM can output the additional cost and schedule required to complete entire project scope informed by issue-level detail

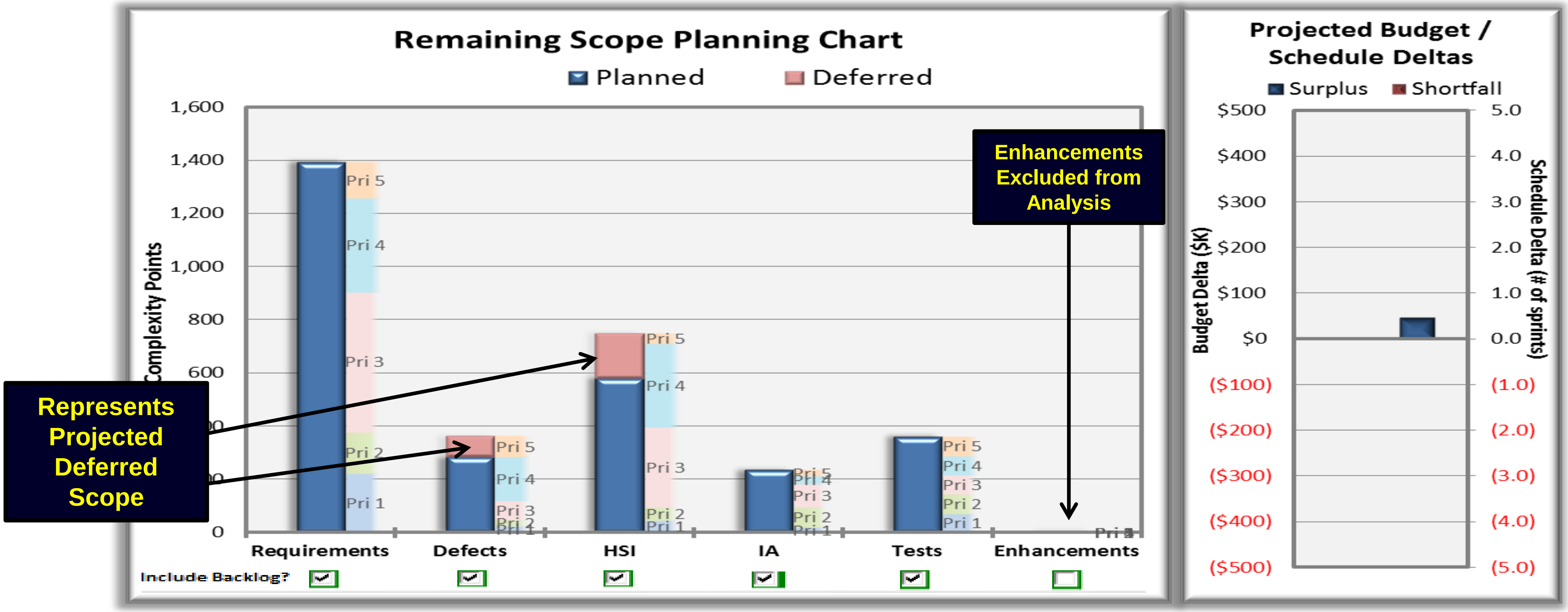




- In order to complete the entire scope on the project, ACME requires an additional $1.3M above current funded value and an additional 4 sprints.

Budget Delta ($K) - Surplus / (Shortfall) :	($1,323)
Schedule Delta (# sprints) - Surplus / (Shortfall) :	(3.7)

Constraints on the project budget and schedule require ACME to prioritize remaining effort/scope



Budget Delta ($K) - Surplus / (Shortfall) :	$1
Schedule Delta (# sprints) - Surplus / (Shortfall) :	0.5

- ACME and the Government have agreed to focus on addressing all Requirements, Tests, and IA Issues and Pri 5 Defects, and some Pri 4 and all Pri 5 HSI Issues

Can dynamically updates a list of specific scope items projected to be deferred as changes are made to the “Remaining Scope Planning Chart”

List of Defect Backlog Artifacts Deferred from Current Project

#	Artifact Code	Priority	Artifact Short Description
1	artf118868	5	Report doesn't generate properly when . . .
2	artf239908	5	The system crashes when . . .
3	artf232404	5	Incorrect data is displayed when . . .
4	artf135259	5	Input form doesn't launch when . . .
5	artf118864	5	Report doesn't generate properly when . . .
6	artf159298	5	The system crashes when . . .
7	artf222810	5	Incorrect data is displayed when . . .
8	artf206800	5	Input form doesn't launch when . . .
9	artf236958	5	Report doesn't generate properly when . . .
10	artf238840	5	The system crashes when . . .
11	artf239918	5	Incorrect data is displayed when . . .
12			
13			
14			
15			

Count of Dropped Artifacts: 11

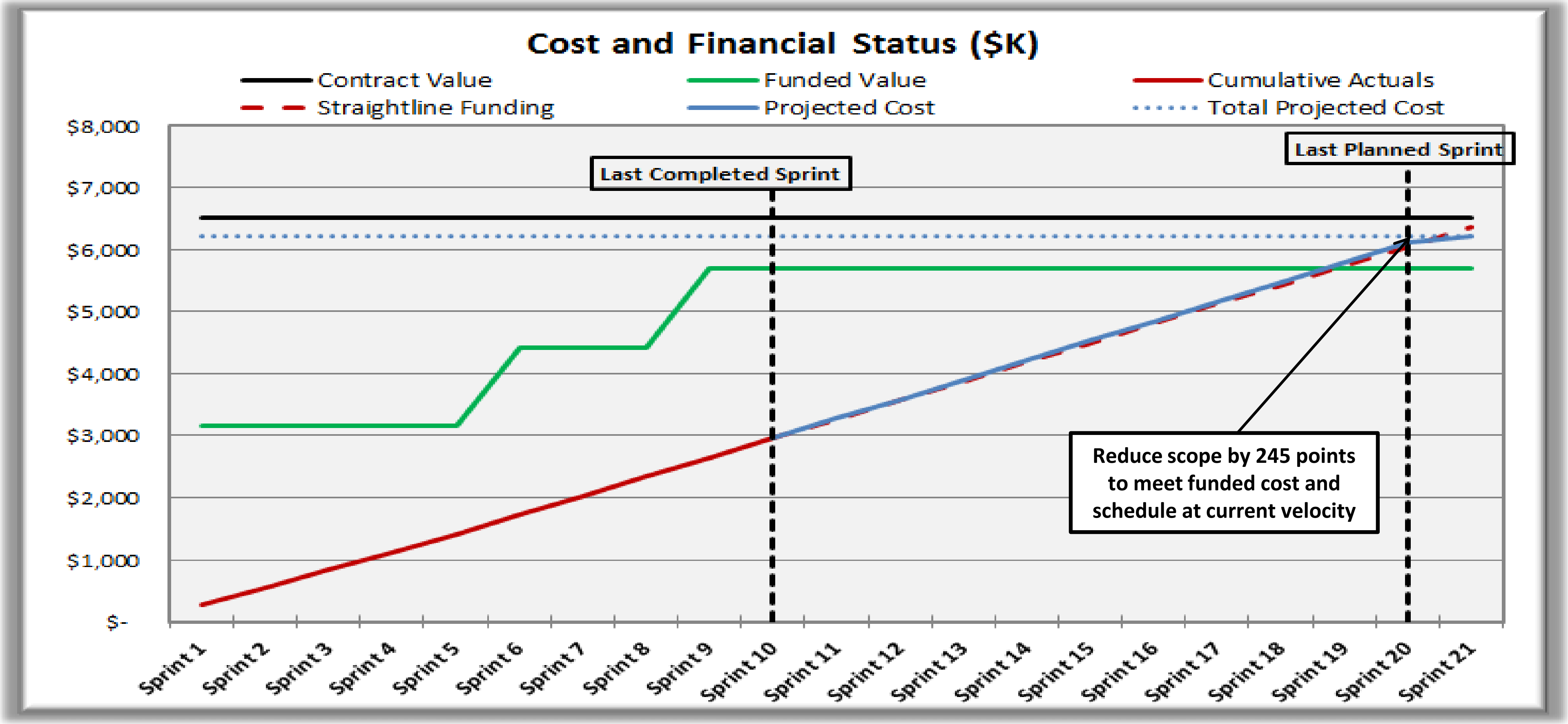
List of HSI Backlog Artifacts Deferred from Current Project

#	Artifact Code	Priority	Artifact Short Description
1	artf224149	4	User needs to enter PIN too many times.
2	artf225224	4	It takes too long to fill out form XYZ.
3	artf225217	4	Process to fill in user input for XYZ is non-intuitive.
4	artf222396	4	Text on page XYZ is too small . . .
5	artf222371	4	Button XYZ should be on the left vs. the right . . .
6	artf222368	4	User needs to enter PIN too many times.
7	artf222387	4	It takes too long to fill out form XYZ.
8	artf215730	4	Process to fill in user input for XYZ is non-intuitive.
9	artf224144	4	Text on page XYZ is too small . . .
10			
11			
12			
13			
14			
15			

Count of Dropped Artifacts: 9

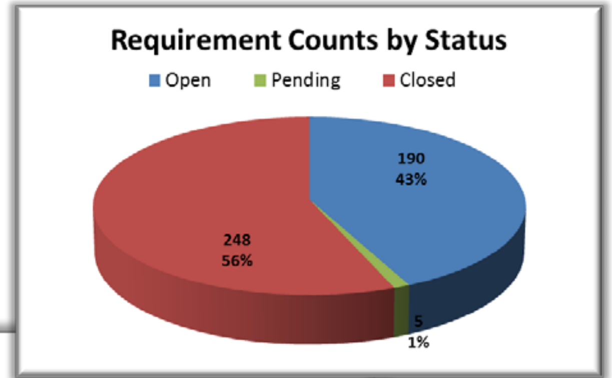
Insight into analysis on actual and projected costs, contract and funded values, and scope variability

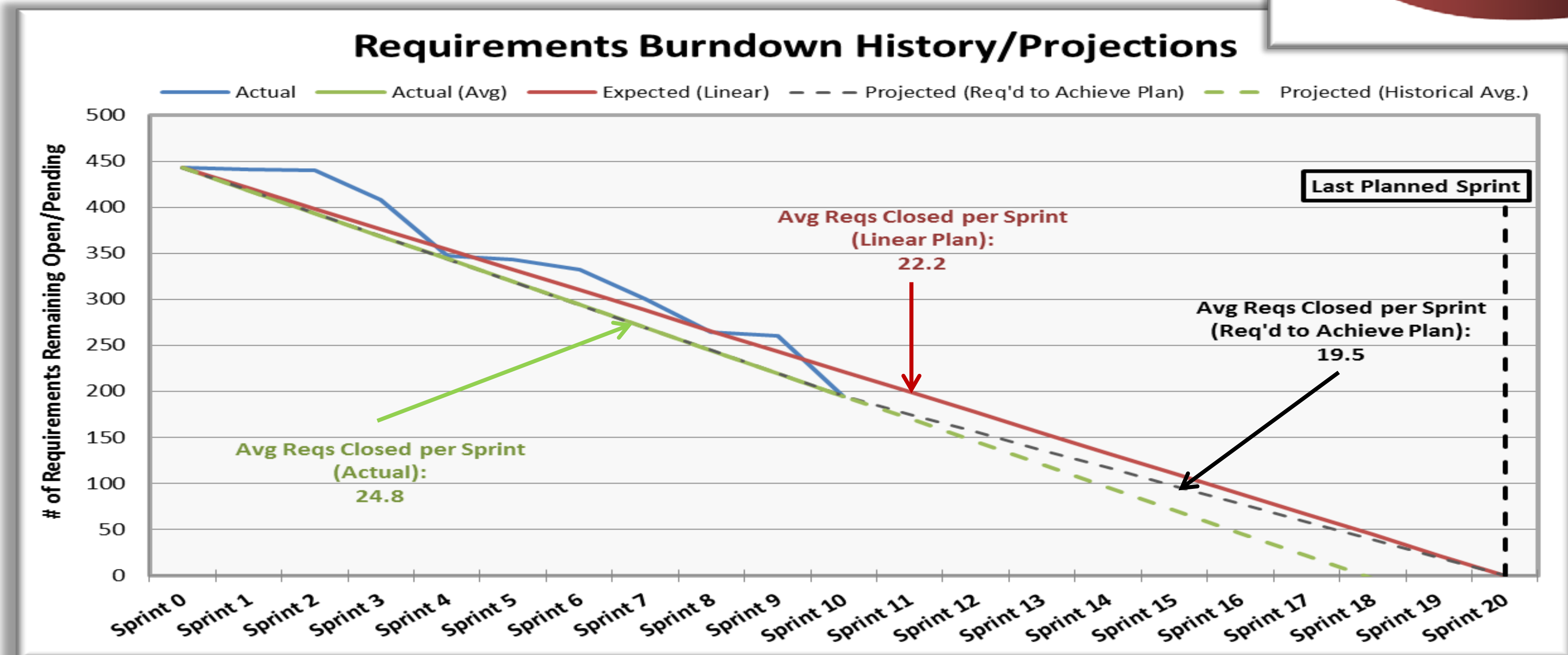
As of Sprint 10 ($)	Contract Value	Funded Value	Actuals Thru 30 Sep 2014	Percent Contract - Funding	Percent Spent - Funding	Percent Spent - Contract Value	Percent Complete - Requirements
Total	**$6,500,000**	**$5,705,000**	**$2,954,000**	**88%**	**52%**	**45%**	**57%**



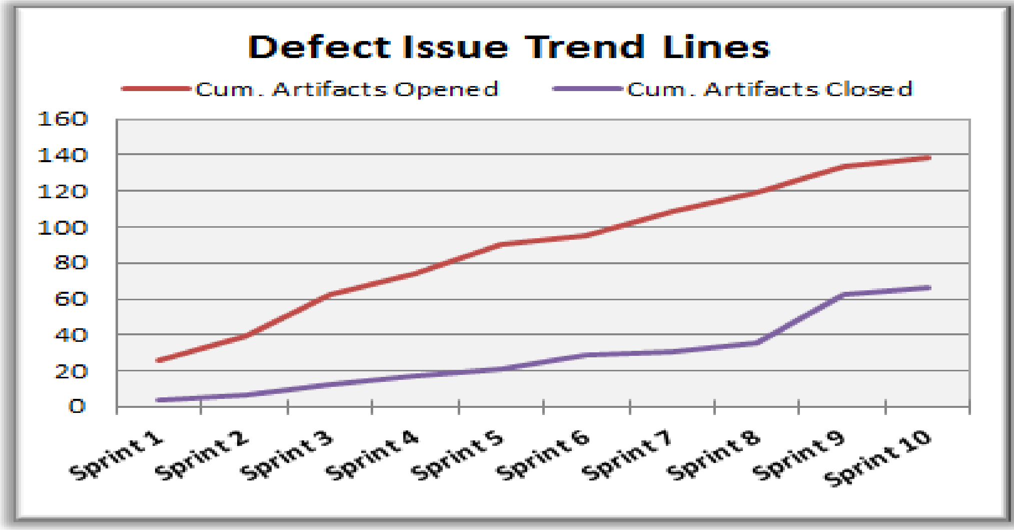
Requirements status, actual burndown, and completion projections are key measures to evaluate current and projected project success

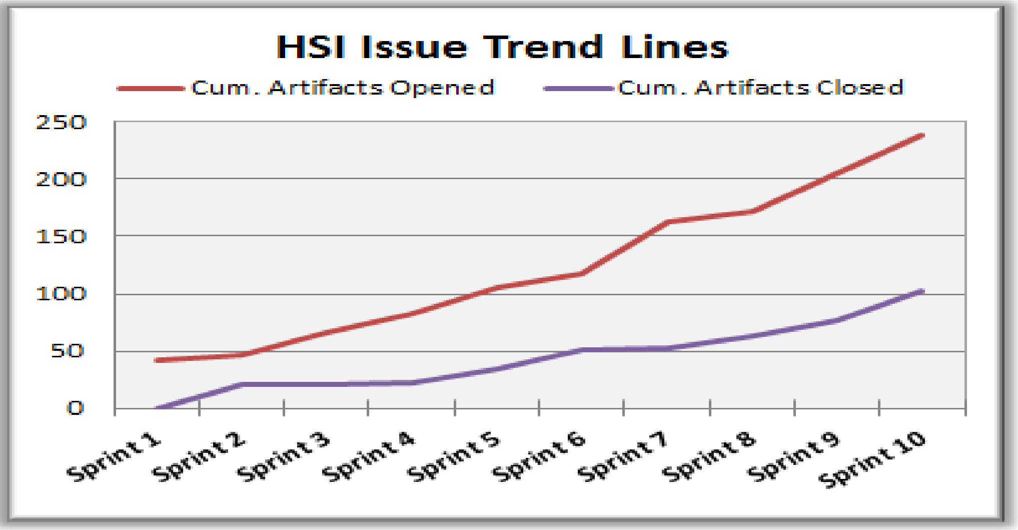
- Requirement Status as of Sprint 10
  – 56% (248 reqs) have been completed
  – 43% (190 reqs) are in open status
  – 1% (5 reqs) are awaiting gov't acceptance testing

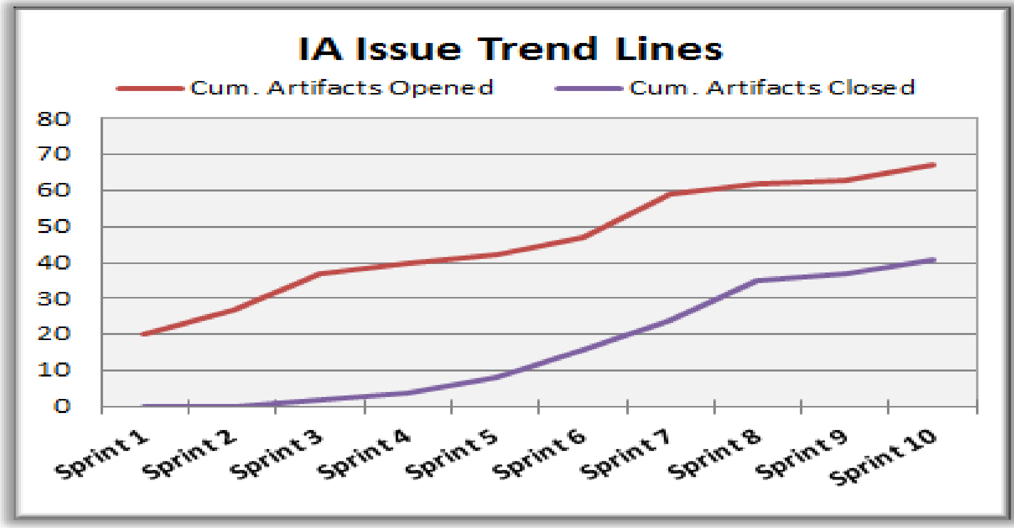


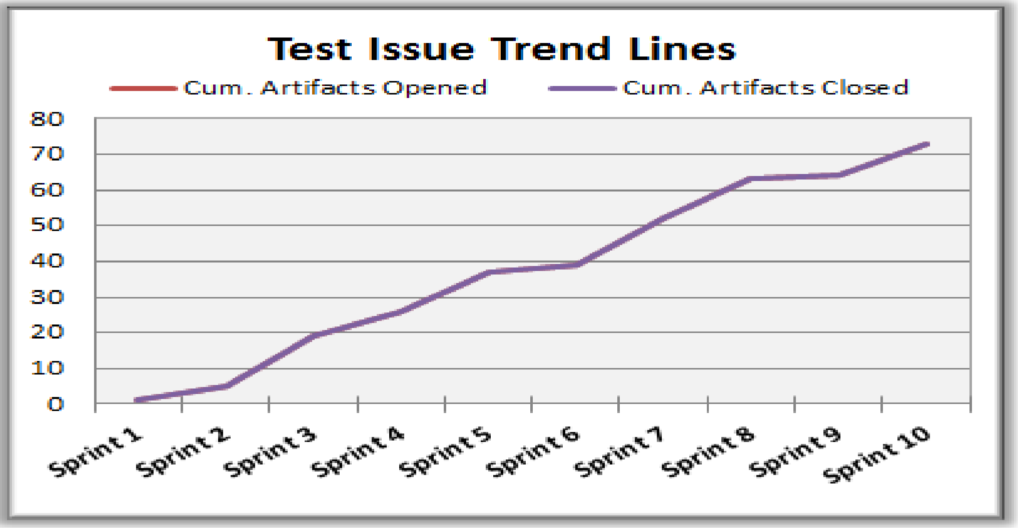


Issue trend line charts provide insight into status of all issue types, providing more detailed insight into problem causes than EVM

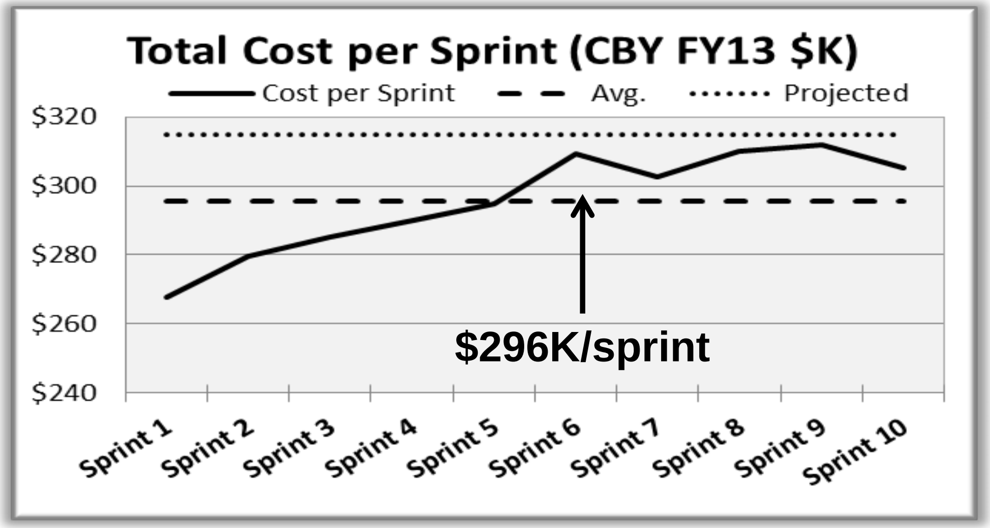


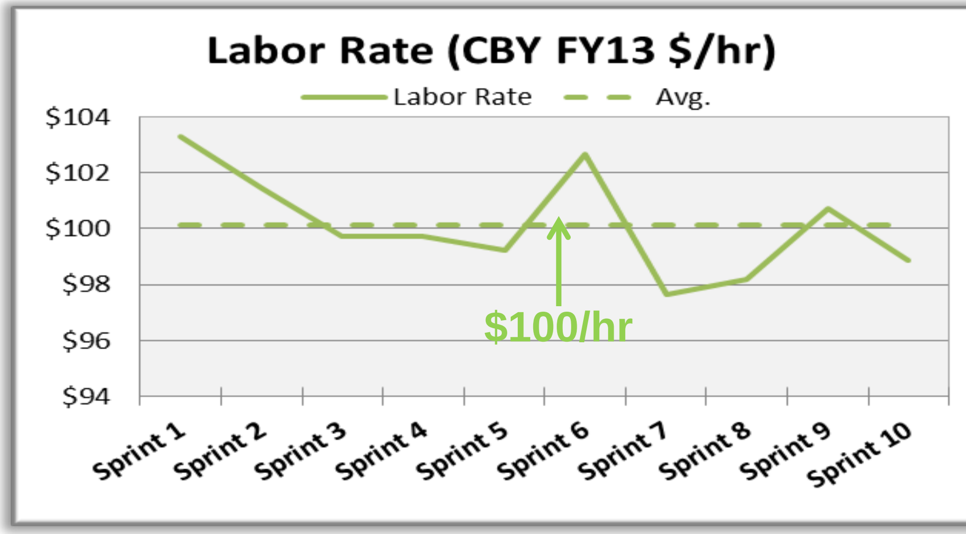


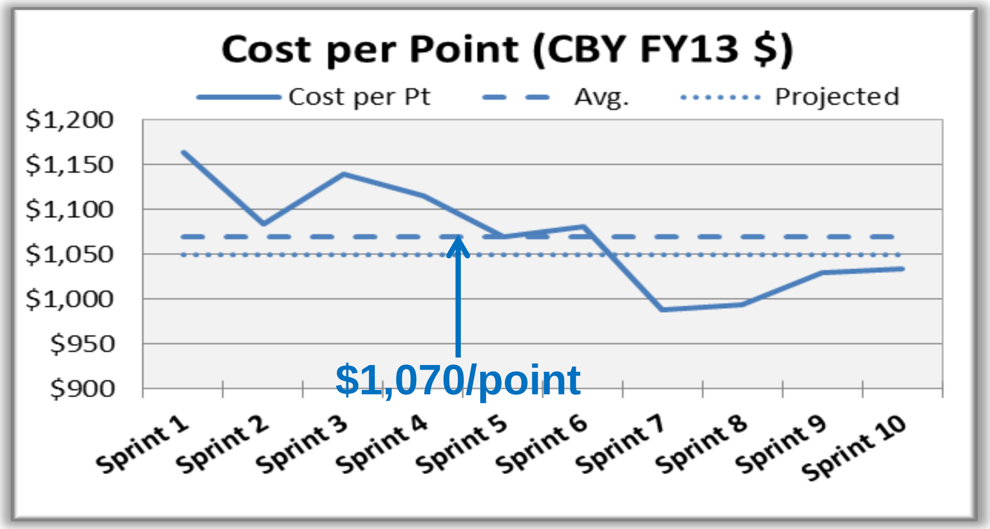


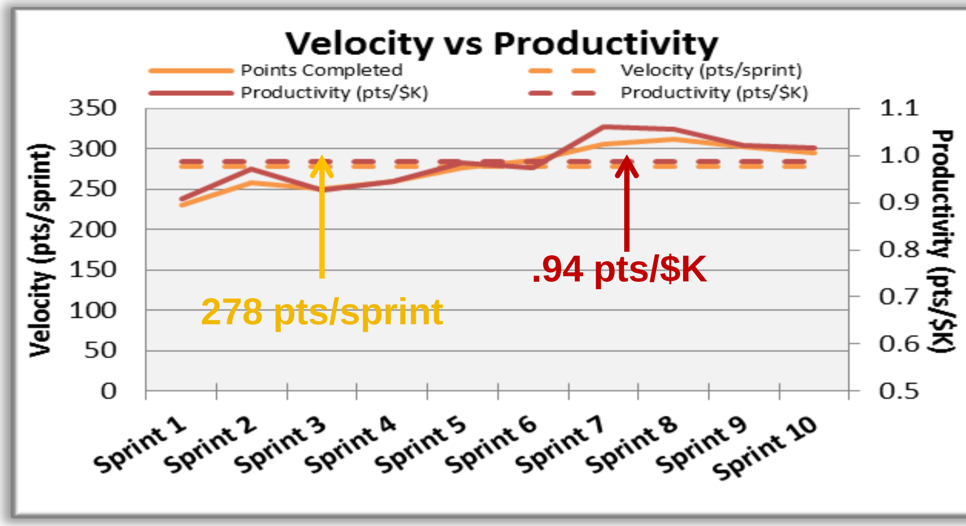


Provide detailed visualizations of historical performance indicators specific to an agile project

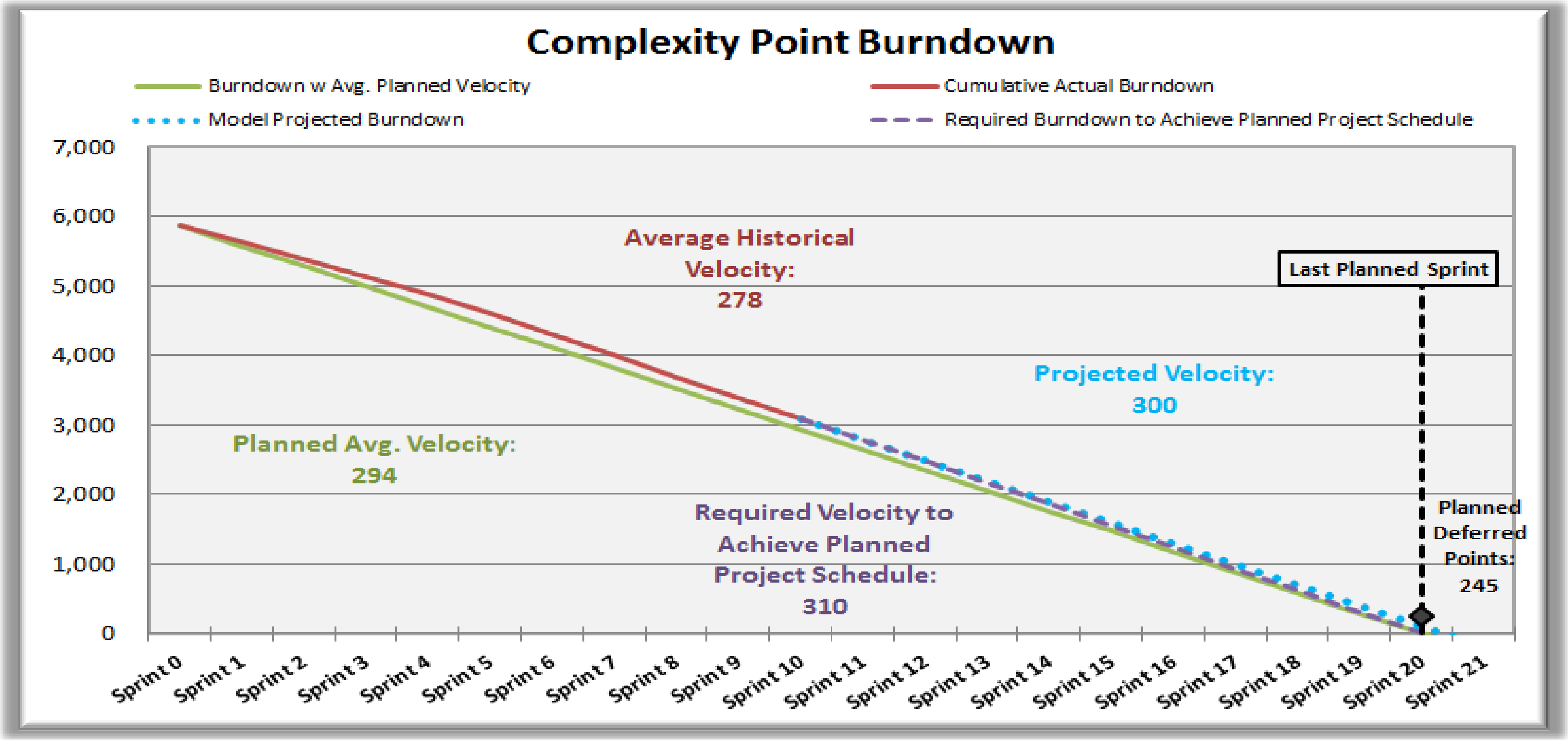




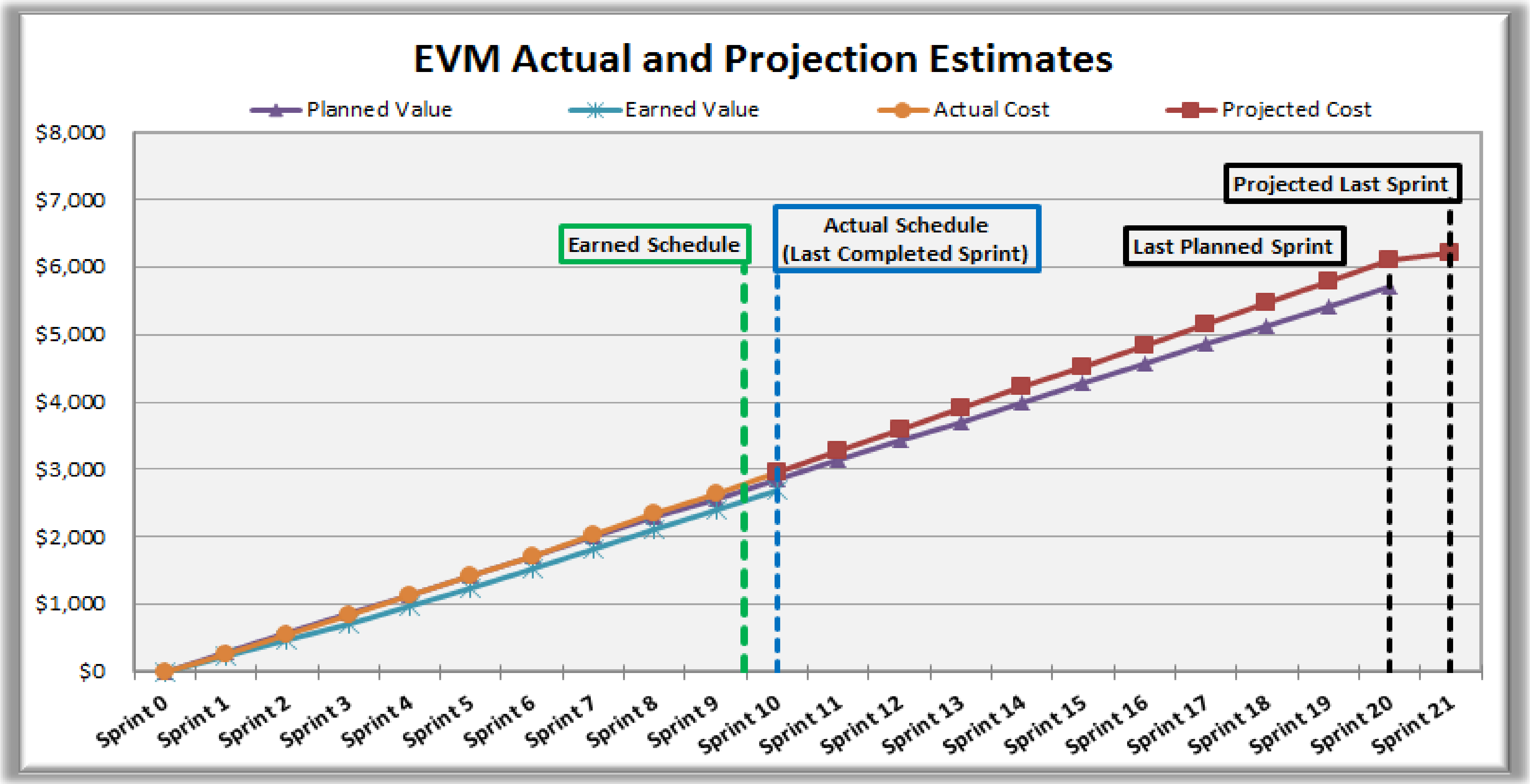




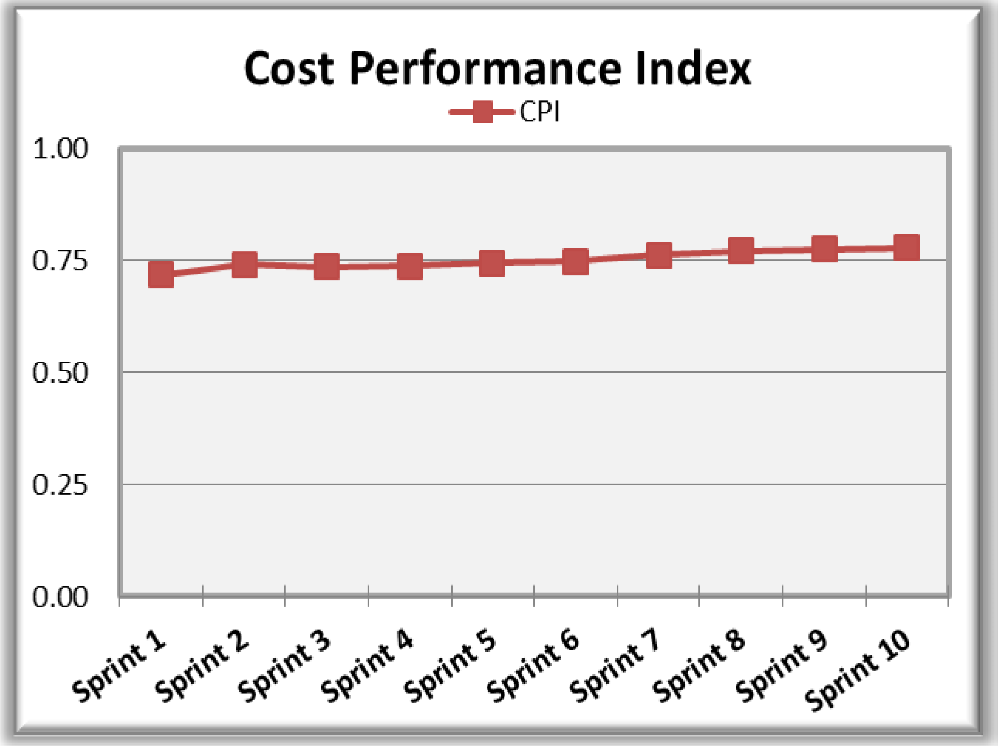
Complexity point burndown analysis allows for trends to be identified and projections to be established

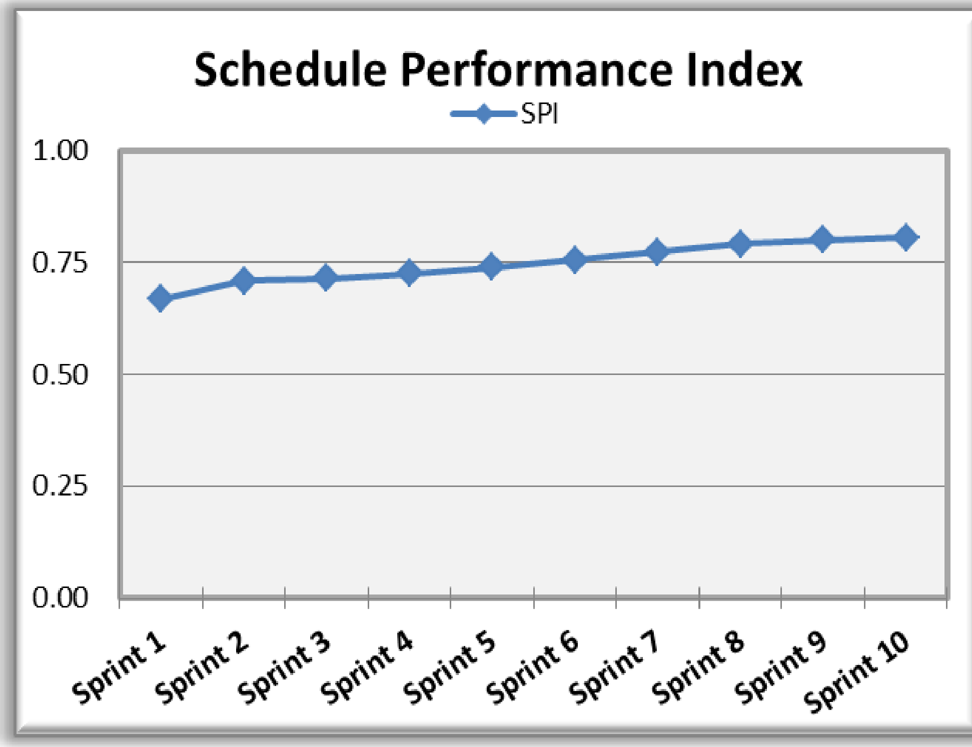


Actual and projected EVM metrics as PV, EV, AC, ES, and Projected Cost for each sprint



Also can provide insight into other EVM metrics





Variance and Performance Indices

Cost		Schedule	
CV	($656)	SV	($554)
CV %	(23%)	SV %	(19%)
CPI	78%	SPI	81%

EVM Metrics

PV	$2,853	ETC_{ML}	$4,378	BAC	$5,705	APC	40%
EV	$2,298	ETC_{WC}	$5,434	VAC	($1,628)	EAC_{ML}	$7,333
AC	$2,954	ECD	11/5/2015	VAC %	-29%	EAC_{WC}	$8,388

Discussion Topics

- Agile Intro
- Challenges Implementing EVM on Agile SW Dev. Projects
- Comparing EVM and an Application Lifecycle Management (ALM) Tool
- Summary
- Q&A

Metrics obtained via an ALM can provide more meaningful metrics than can be obtained through EVM and at a fraction of the cost

- **Project Baseline goes against Agile Tenants –** Progress should be tracked and managed as establishing a project baseline rarely ever occurs at Sprint 0
- **Insight into status at the Artifact/Issue level** – Allows for effective decisions to help the product owner prioritize remaining effort
- **Software Growth as a result of Technical Debt** – Tradeoff analysis to determine whether to focus on Requirements, HSI, Defects, and/or others
- **Typical Implementation of EVM can require 1-2 FTEs** – Using data that already exists in an ALM can reduce effort and cost by a magnitude if not more
- **More Meaningful Metrics** – Metrics are understood by Developers and project stakeholders
- **ANSI-748 Compliance can be Restrictive** – Not conducive to an Agile environment, high maintenance costs, implementation requirements, constant need to rebaseline

Discussion Topics

- Agile Intro
- Challenges Implementing EVM on Agile SW Dev. Projects
- Comparing EVM and an Application Lifecycle Management (ALM) Tool
- Summary
- Q&A

Q&A

For further information . . .

Omar Mahmoud
President

Barakah Consulting
Tel 714.421.1231
omar.mahmoud@barakahconsulting.com

Measuring Benefits of Innovation

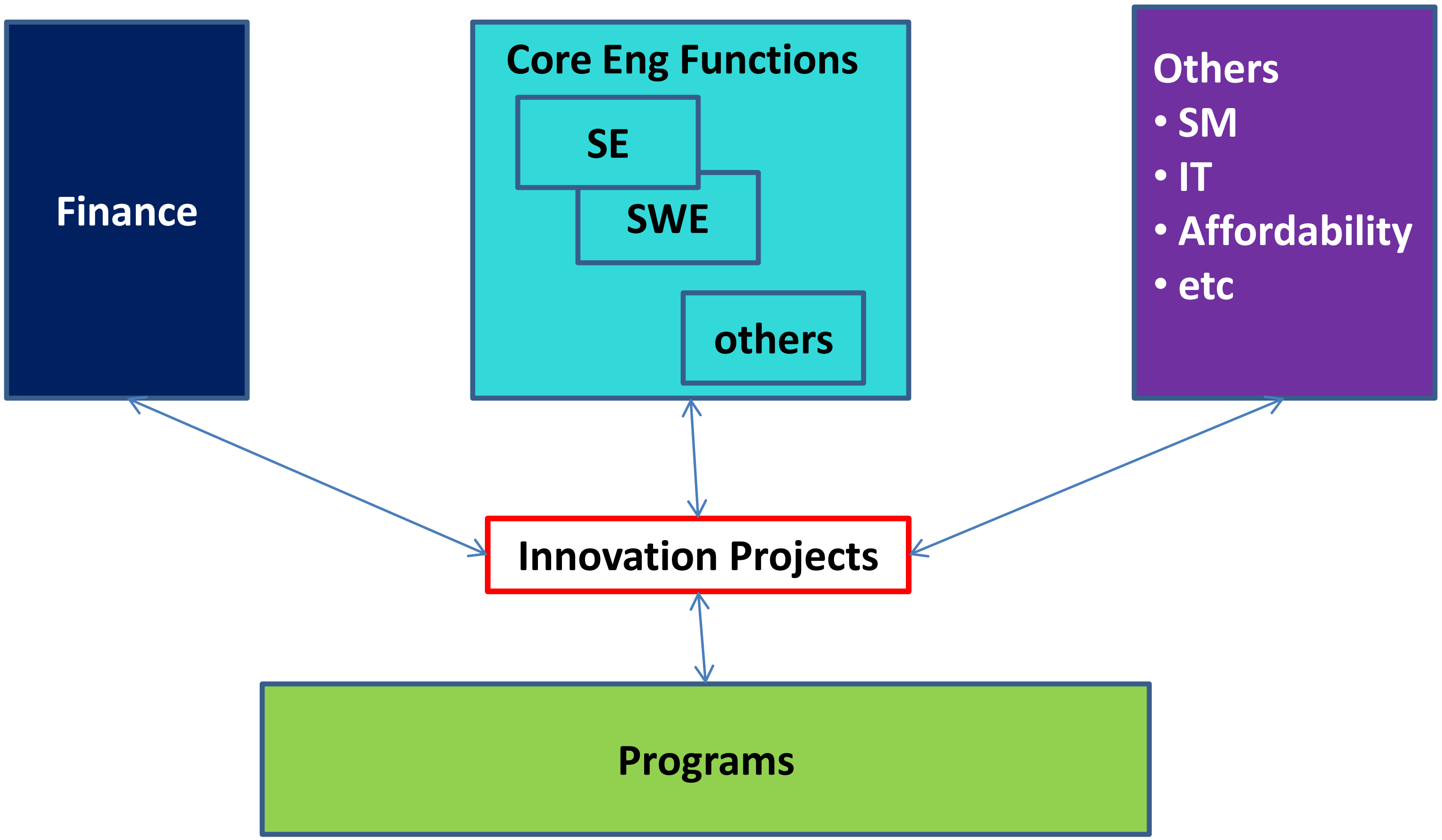
Steve Shyman
Shawn Rahmani
Boeing Defense Space/Security (BDS)
August 11, 2015

Outline

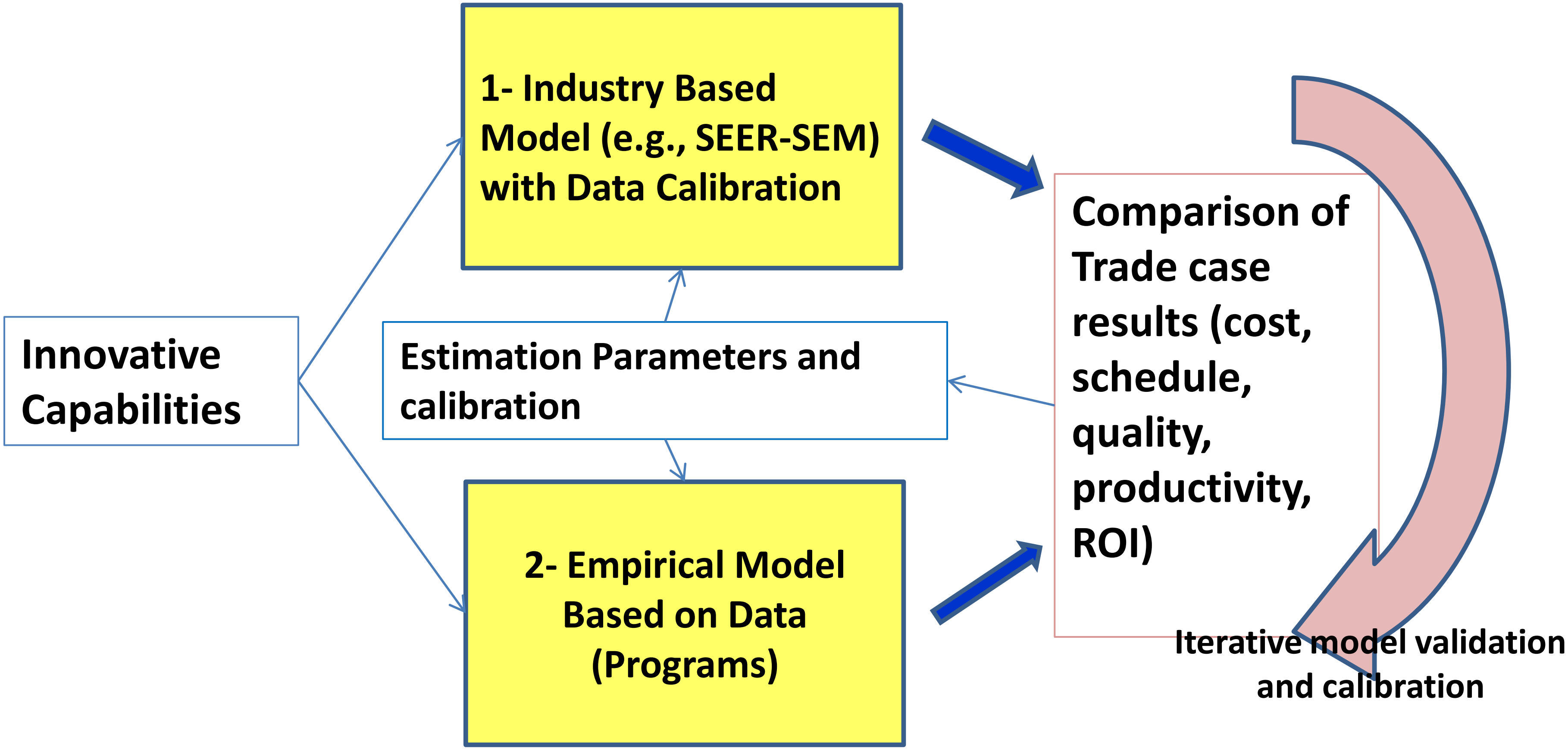
While we know we are improving productivity over time tying improvements to specific innovations is difficult…multiple approaches combined can provide innovation benefit validation

- **Introduction**
- **Measuring Productivity Improvement**
- **Approach as Applied to Agile**
- **Results**
- **Summary**

Organization



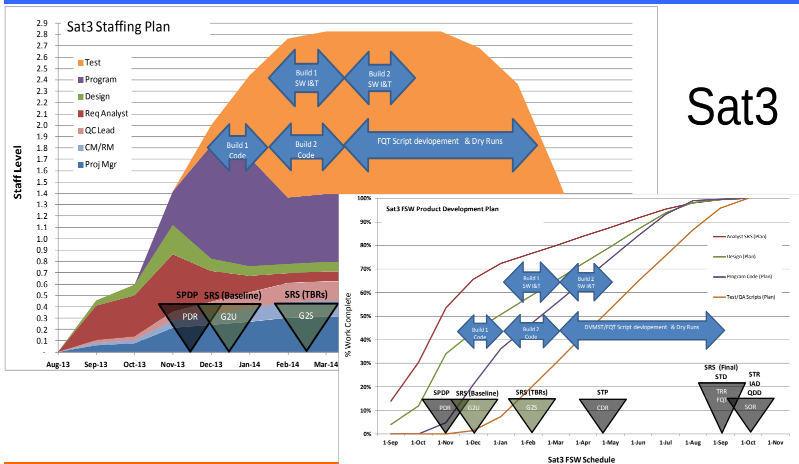
Innovation Modeling for Measuring Benefits



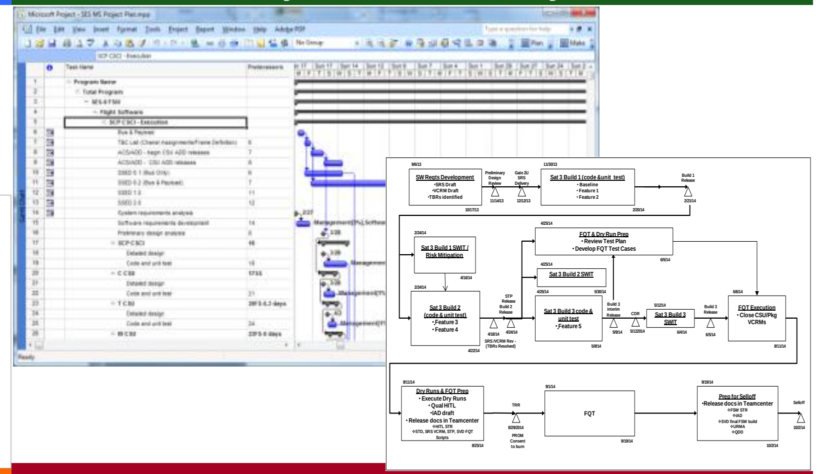
Validated Program Data Is Key

Examples of Innovation Measurement Artifacts

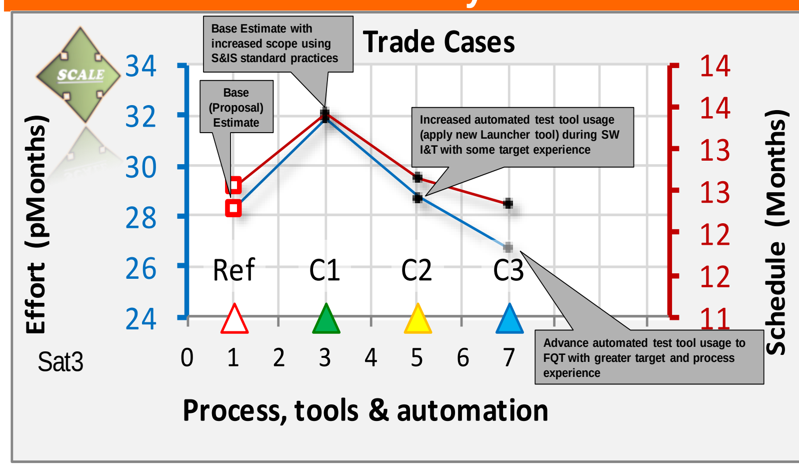
Detailed Staffing Plan / % Work Product Complete



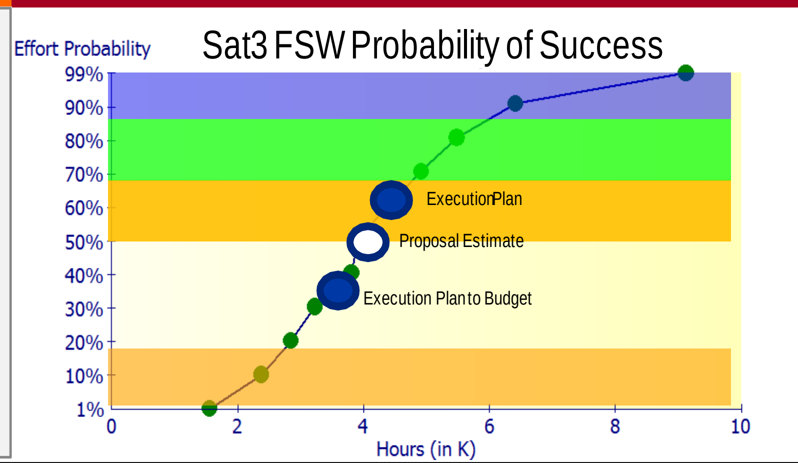
MS Project Plan/Activity Flow



SW Affordability Trade



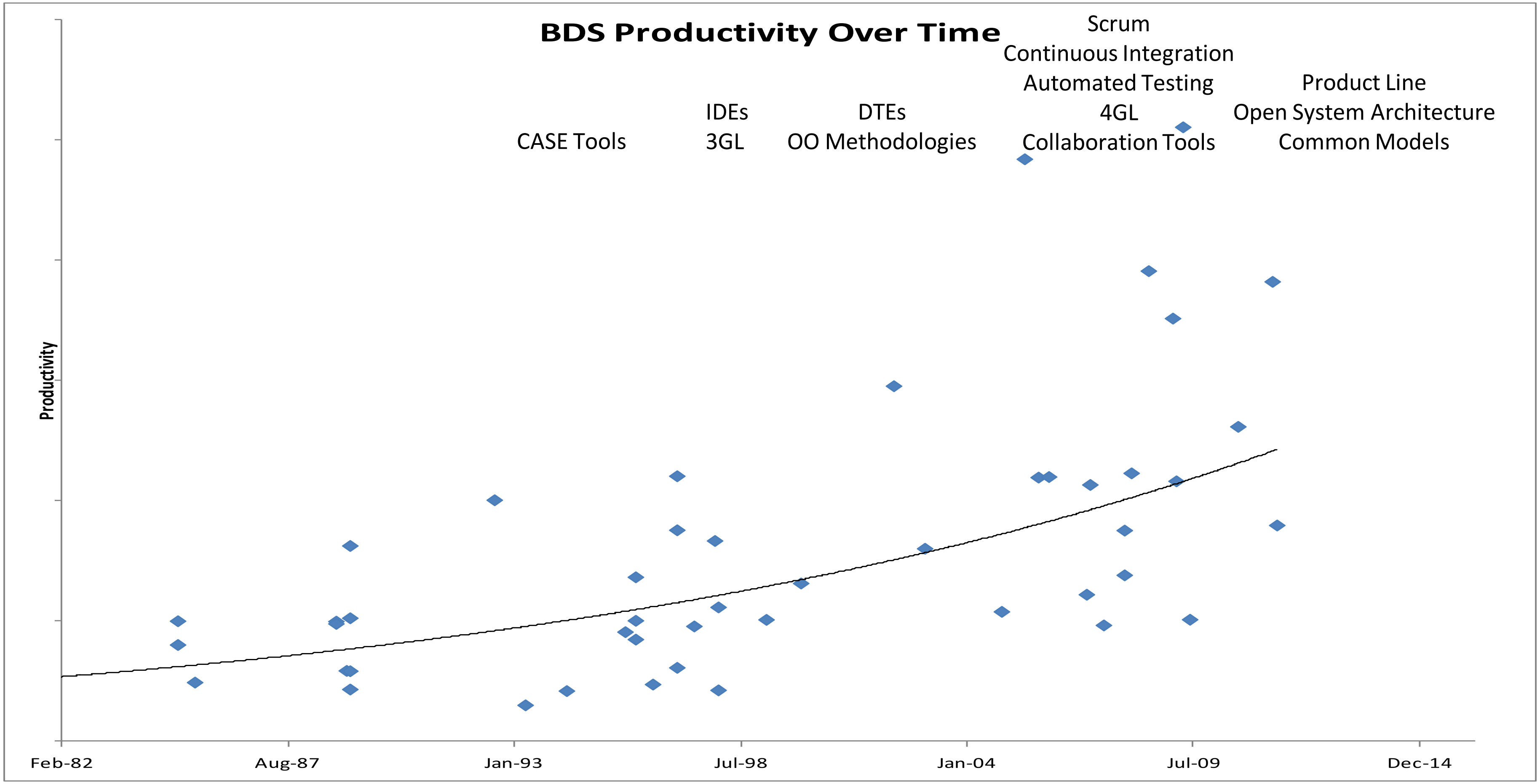
Risk Assessment



Measuring Innovation-Driven Productivity Improvement

- **Controlled experiments are not conducted**
- **Recognize many factors that can influence productivity**
  - **Personnel, Complexity, Standards…**
- **Examining successive increments of a product provides some control over above factors but not perfect**
  - **Experience, Stability, Integrated product…**
- **Lifecycle productivity benefits may be difficult to measure when some benefits are indirect**
  - **E.g. Applying a requirements quality improvement effort that may increase the system engineering effort but reduce the downstream design, integration, and test efforts**

BDS Continuous Incorporation of State of the Art Technology



- **Trend shows increasing productivity through the years**
- **Data points represent wide variety of platforms and applications**

CASE – Computer Aided SW Engineering
IDE – Integrated Development Environments
DTE – Desktop Test Environment
GL – Generation Language
OO – Object Oriented

Data Consistency

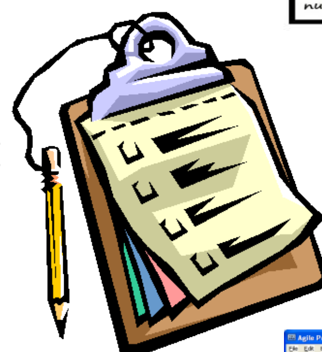
When comparing productivities across projects need to ensure consistent measurement to avoid differences due to measurement variances

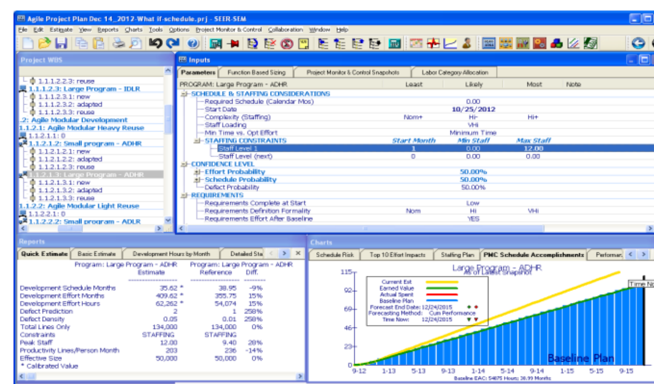
- **Standard size measurement**
  - **Standard definition**
  - **Standard tool for counting**
  - **Process documentation for guidance**
- **Standard effort measurement**
  - **Standard task based accounting across enterprise**
- **Standard productivity measure**
  - **Standard definition of what effort is included**

Approach to Measuring Benefits of Innovation Agile Example

- **Three prong approach**
  - **Collect *hard* actual data from programs and analyze**
  - **Survey programs for expert *judgment* data**
  - **Calibrated model based trade studies**

Trial	DATA: Brand X – number of chips in 3 cookies	Brand Y – number of chips in 3 cookies
1	14	16
2	16	17
3	17	18
4	21	16
5	8	14
total number	76	81
number per cookie	15.2	17.2



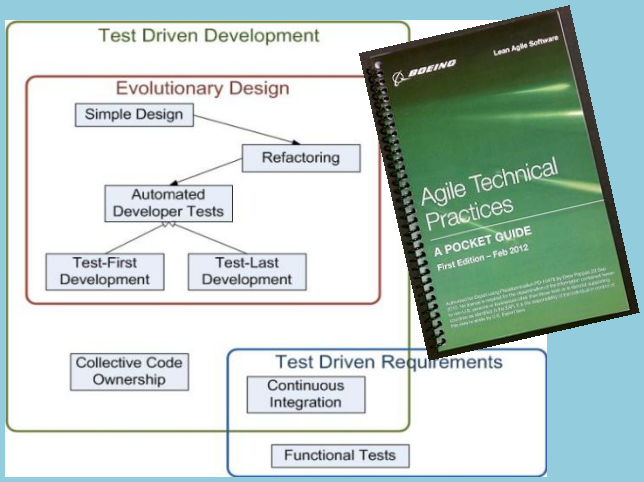


The Boeing Agile Software Process (BASP) is:

Agile Project Management + Agile Technical Practices

Scrum

Automated Testing & Continuous Integration

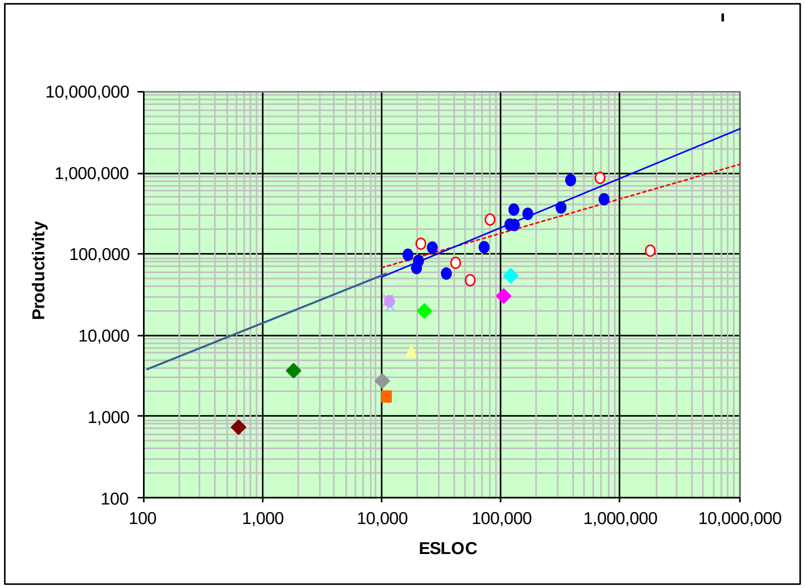


http://basp.web.boeing.com/

http://basp-atp.web.boeing.com/

Hard Data Collection

- *Hard* data implies data is auditable
- While many projects are implementing Agile at Boeing…
  - Many follow-on and maintenance type programs
  - Many efforts not complete yet
  - Only part of project implementing
  - No reasonable control example exists



- Data shows less hours spent to produce eSLOCs for the Agile software*
- Blue dots are non-Agile projects
- Red dots are outliers

* Caveat: Agile programs differ from the non-Agile either in application or program phase or measurement methods

Hard Data Collection, continued

- Left with few anecdotal examples that indicate significant productivity improvements
  - Avionics program, two CSCIs where one implemented Agile in third iteration and the other did not
    - Agile effort doubled productivity from 2nd to 3rd iteration, but…
    - Non-agile effort also realized significant productivity gains
    - Difficult to distinguish benefits from follow-on vs Agile effects
  - Ground mobile, modification program implemented Agile
    - Productivity more than doubled, but…
    - Original program suffered many problems including integration with suppliers of new technology
    - Modification program small in comparison and without major integration of new technology
    - Utilized new team, brought suppliers in-house

Expert ***Judgment*** Data

- **Lacking enough hard data for significant statistical analysis turned to surveying Agile teams for estimated cost reductions from implementing Agile**
  - **Surveyed over 100 teams within Boeing**
  - **Captured additional key information about project including:**
    - **Size**
    - **New vs Follow-on**
    - **Technical Practices**
    - **Personnel**
- **Overall results showed on average a 24% cost reduction for Agile programs**
  - **Caveat is that additional factors may have contributed**
  - **Based on judgment of software managers/leads utilizing whatever data they have at hand**

Expert *Judgment* Data, Additional Analysis

- **Examining segregated data sets based on additional characteristics collected more details emerged**
  - **Better personnel more reduction**
  - **Use of technical practices increased reduction**
  - **Small programs had more reduction**

		With/Without Automated Testing & Cont. Integration (All)					With Automated Testing and Continuous Integration					Without Automated Testing & Cont Integration				
Size	Personnel Capabilities	No Obs	Min	Average	Max	Std Dev	No Obs	Min	Average	Max	Std Dev	No Obs	Min	Average	Max	Std Dev
All	All	100	-3%	24%	78%	19%	58	4%	28%	78%	21%	29	-3%	18%	66%	16%
	Above Average	67	5%	28%	78%	21%	39	5%	33%	78%	23%	22	5%	20%	66%	17%
	Average	32	-3%	18%	42%	11%	18	4%	17%	42%	12%	7	-3%	11%	25%	9%
	Below Average	1	11%	11%	11%	N/A	1	11%	11%	11%	N/A					
Small	All	50	5%	31%	78%	23%	28	5%	38%	78%	25%	18	5%	22%	66%	18%
	Above Average	34	5%	38%	78%	24%	20	5%	47%	78%	24%	13	5%	26%	66%	19%
	Average	16	5%	16%	42%	11%	8	5%	17%	42%	13%	5	5%	11%	15%	5%
	Below Average															
Medium	All	19	4%	17%	39%	11%	12	4%	17%	39%	12%	4	5%	15%	25%	12%
	Above Average	12	5%	20%	39%	11%	6	5%	22%	39%	13%	4	5%	15%	25%	12%
	Average	6	4%	14%	27%	10%	5	4%	12%	27%	9%					
	Below Average	1	11%	11%	11%	N/A	1	11%	11%	11%	N/A					
Large	All	31	-3%	18%	44%	11%	18	5%	18%	38%	10%	7	-3%	10%	25%	9%
	Above Average	21	5%	16%	44%	11%	13	5%	16%	35%	9%	5	5%	9%	15%	5%
	Average	10	-3%	22%	38%	12%	5	15%	24%	38%	12%	2	-3%	11%	25%	20%
	Below Average															

Model Based Analysis

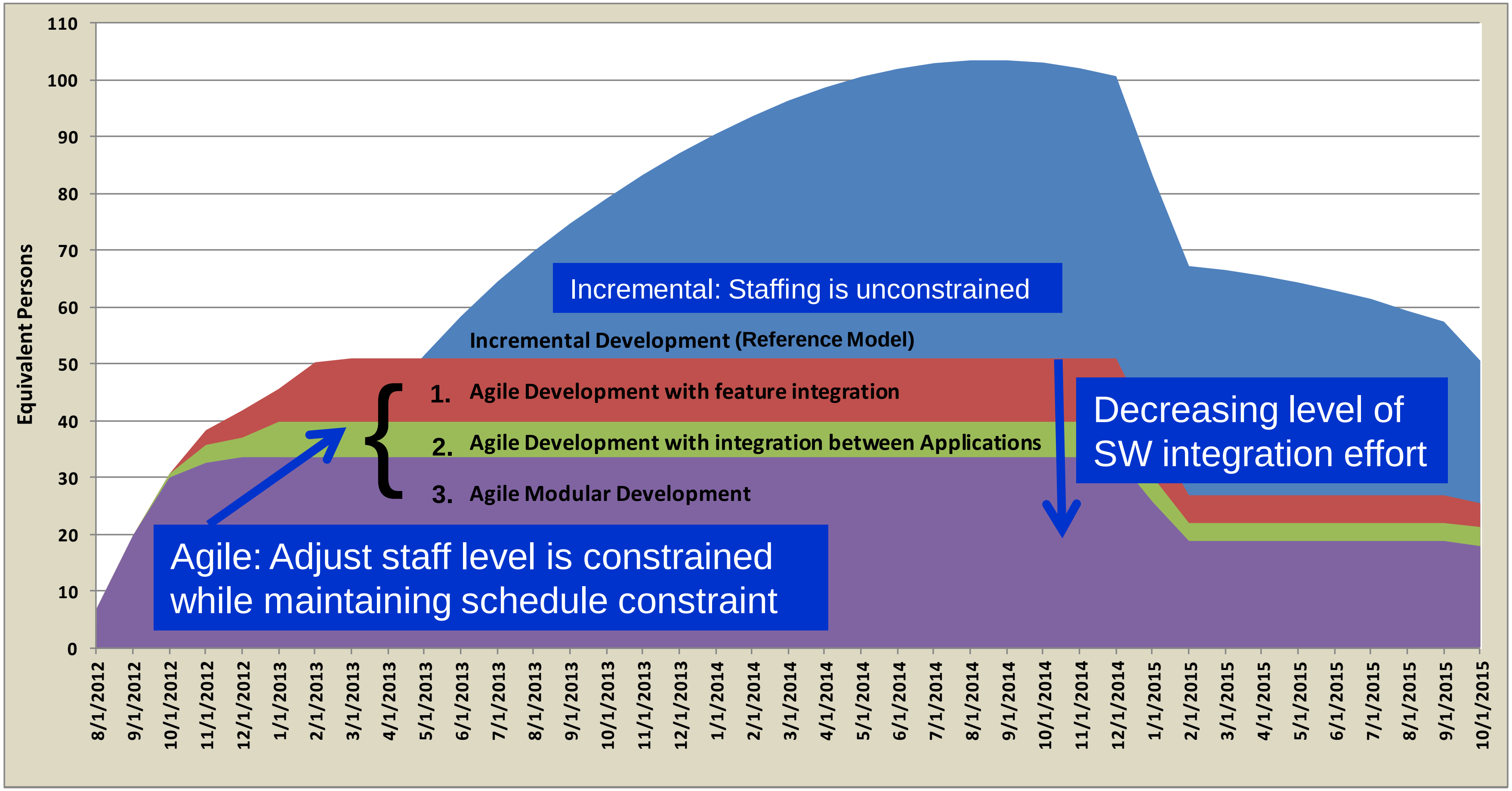
- **Utilize SEER-SEM from Galorath Inc. as underlying engine or model in analysis**
  - **Additional Boeing developed (or developed specially for Boeing) spreadsheet tools for interaction with SEER-SEM**
  - **Calibrate:**
    - **Overall productivity by adjusting parameter inputs**
    - **Allocation of labor across tasks and phases using Labor Category Allocation tables and calibration factors**
      - Requirements, Design/Code/Test, CM, QA
    - **Staffing profiles**
  - **Create Custom Class Knowledge Bases**

Model Based Analysis Applied to Agile

- Identified key characteristics of Agile that believed led to cost reductions
  - Small team, modular developments
  - Frequent iterations
  - Technical practices
- Aligned characteristics to model inputs
  - Programs Included in Size to represent teams or features
    - Offset by use of Programs Concurrently Integrating
  - Staffing Constraints to control team size, peak staffing level and staffing profile in general
  - Automated Tool Usage for technical practices

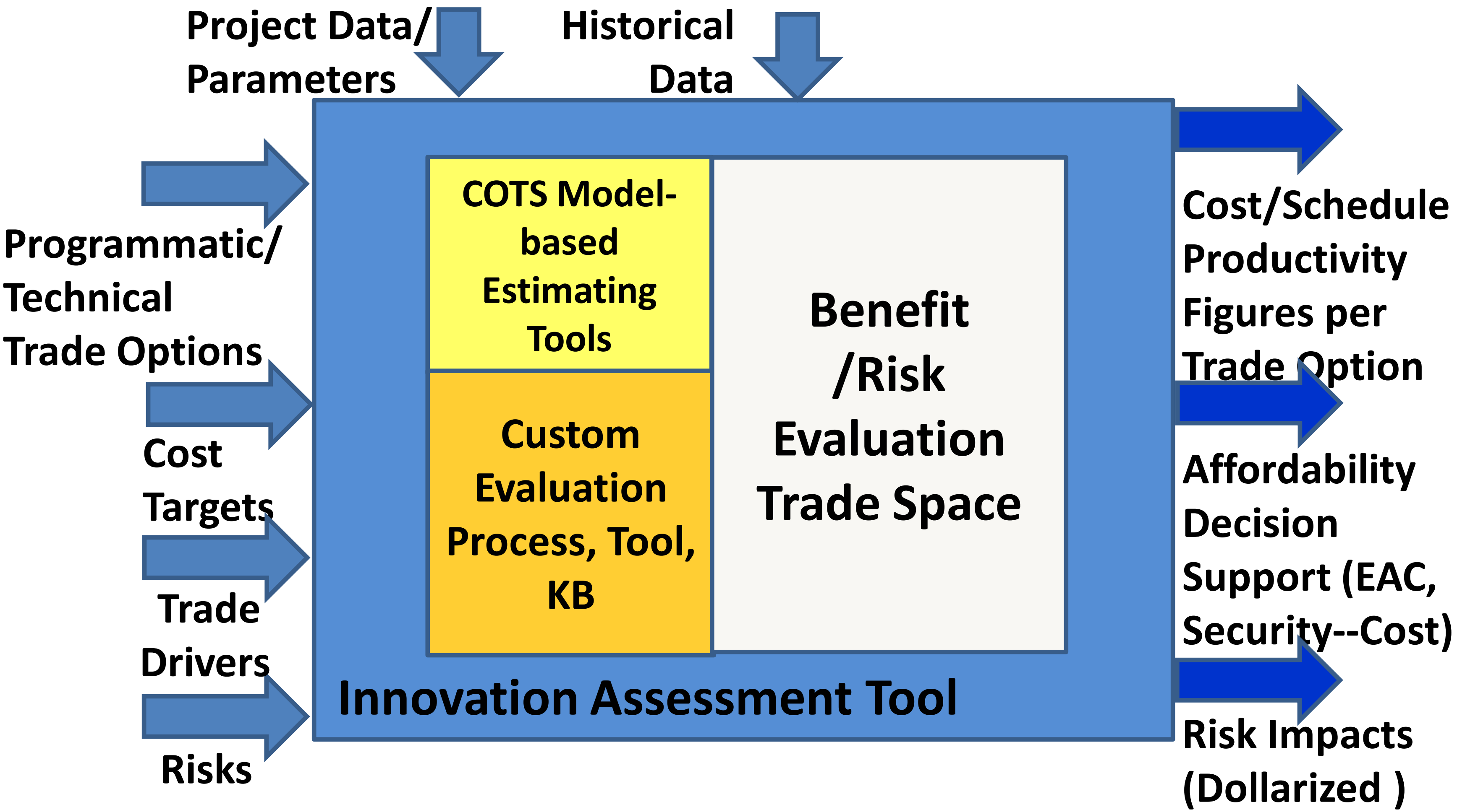
Model Based Approach Results

Incremental vs. Agile with varying degrees of integration effort



Analysis suggest potential for significant productivity improvement, anywhere from 15-50%, depending on architecture and other factors.

Framework for Model Integration, Innovation Assessment and Benefit Trades



Software Focused, Extendable to Systems Engineering

Merging the Results

- ***Hard*** **data suggest significant improvement but…**
  - **Difficult to assess direct cause and effect**
  - **Lacks in numbers for statistical analysis**
- **Expert** ***Judgment*** **data suggests significant improvement**
  - **Based on large population surveyed but…**
  - **Lacks actual data for verification**
- **Model based analysis captures the ‘why’ characteristics of the innovation and demonstrates potential improvements**
  - **Model based on industry data, probably limited in Agile data**

That the three approaches imply similar conclusions validates the benefit of the innovation and mitigates the perceived risk as applied in proposals.

Summary

- **Estimating quantitative benefits of innovations requires**
  - **Framework for data collection and measurements**
  - **Development of model-base techniques using measurements/data**
- **Collection of "valid" data (measurements) is challenging**
- **Models using simpler empirical data can be a good start**
  - **Agile-based productivity measurements**
- **Improved data collection and model calibration is needed**
  - **Standards, automation, and mandated policies can help**

Increased customer and contractor interaction for data analysis:

- **Greater understanding of the data and causes of variance**
- **Innovation analysis for more optimized capability**

ASSIGNMENT OF COPYRIGHT
Publisher: Practical Software and Systems Measurement (PSM) Users' Group Workshop
And Software and Information Technology Cost IPT ("Publisher)

Work Title: Measuring Benefits of Innovation ("Work")

Boeing Employee(s)/Author(s): Steve Shyman, Shawn Rahmani

At least one, but possibly not all, of the authors of the Work is an employee of The Boeing Company or one of its wholly-owned subsidiaries ("BOEING") who prepared the Work within the scope of his or her employment. The Boeing employee(s) listed above are not authorized to assign or license Boeing's rights to the Work, or otherwise bind Boeing. However, subject to the limitations set forth below, Boeing is willing to assign its copyright in the Work to Publisher.
Notwithstanding any assignment or transfer to the Publisher, or any other terms of this Agreement, the rights granted by Boeing to Publisher are limited as follows: (i) any rights granted by Boeing to the Publisher are limited to the work-made-for-hire rights Boeing enjoys in the Work; (ii) Boeing makes no representation or warranty of any kind to the Publisher or any other person or entity regarding the Work, the information contained therein, or any related copyright; and (iii) Boeing retains a non-exclusive, perpetual, worldwide, royalty-free right, without restriction or limitation, to use, reproduce, publicly distribute, display, and perform and make derivative works from the Work, and to permit others to do so
.
Nancy Miranda
Contracts Specialist
Intellectual Property Management

COSYSMO 3.0: Updating Cost Estimation of Systems Engineering to Support Affordability

Jim Alstad*	USC Center for Systems and Software Engineering
Dr Barry W Boehm	
Dr Jo Ann Lane	
Garry Roedler	Lockheed Martin
Dr Gan Wang	BAE Systems
Marilee Wheaton	The Aerospace Corporation

Software and IT Cost IPT Meeting

August 11, 2015

Purpose of this Presentation

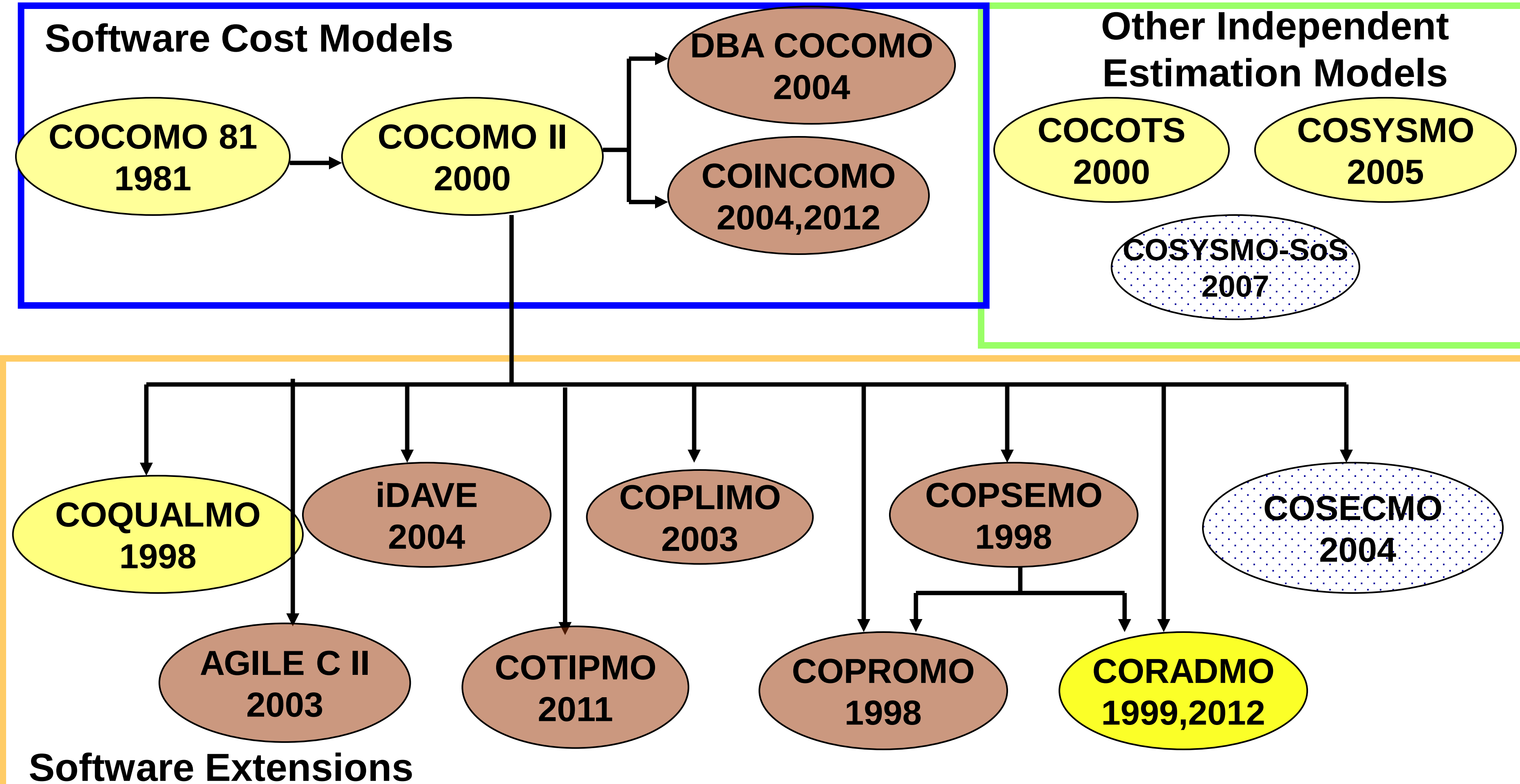
- To introduce people to COSYSMO: the COnstructive SYStem engineering cost MOdel
  - And the effort to develop version 3.0 of the model
- To prepare participants for this afternoon's workshop which will develop COSYSMO 3.0 further
  - Will gather expert opinion on numerical values of parameters in the model
  - Using a structured technique called Wideband Delphi

Agenda

Agenda:

- **History of COSYSMO and COSYSMO 3.0**
- **Overview of “Harmonized” COSYSMO model**

COCOMO Family of Cost Models



Legend:
Model has been calibrated with historical project data and expert (Delphi) data
Model is derived from COCOMO II
Model has been calibrated with expert (Delphi) data

Dates indicate the time that the first paper was published for the model

History of COSYSMO Models

COSYSMO 1.0
Valerdi, 2005

- Identifies form of model
- Identifies basic cost drivers
- Identifies Size measure

With Reuse
Wang et al, 2008

- Adds weights to Size elements, reducing net Size in the presence of reuse

Req'ts Volatile
Pena, 2012

- Adds scale factor based on requirements volatility

For Reuse
Wang et al, 2014

- Adds weights to Size elements, reducing net Size when artifacts are only partially completed

Sys of Sys
Lane et al, 2011

- Adds effort multiplier when in the presence of system-of-systems

COSYSMO 3.0
Alstad, 2016?

- Integrates features of previous models

COSYSMO 3.0 Directions

Harmonize existing COSYSMO family models:

- **Several factors affecting the COSYSMO cost model have been shown to be valuable in increasing estimation accuracy (terminology from [5]):**
  - **Reuse (partial model--SEWR) [3]**
  - **Reuse (with SEFR) [1]**
  - **Requirements volatility (SERV) [4]**

  The rating scales for these could be integrated into a comprehensive COSYSMO model.

Enhancement planned for inclusion:

- **System-of-system considerations are hypothesized to affect system engineering costs:**
  - **Interoperability considerations [6]**

COSYSMO 3.0 Directions Part 2

Enhancements under discussion:

- **Explore a model for total development cost based primarily on the COSYSMO parameters [17, 7]**

Agenda

Agenda:

- **History of COSYSMO and COSYSMO 3.0**
- **Overview of “Harmonized” COSYSMO model**

Harmonized COSYSMO 3.0 Focus of Model

- **The model's focus is a single System of Interest**
  - **I.e., when used in a straightforward way, the resulting estimate is for system engineering costs for one System of Interest based on parameters of that System.**
  - **(A more sophisticated use of a COSYSMO model is given in [10], where Lane shows how to use COSYSMO to estimate, in a system-of-systems context, costs for the SoS overall and for the constituent systems; that work applies to COSYSMO 3.0 as well.)**

Harmonized COSYSMO 3.0 Top-Level Model

$$PM_{C3} = A_{C3} \cdot (Size_{C3})^{E_{C3}} \cdot \prod_{j=1}^{15} EM_{C3,j}$$

Elements of the Harmonized COSYSMO 3.0 model:

- ***Calibration parameter A***
- **Interoperability**
- **Size model**
  - **eReq submodel, where 4 products contribute to size**
  - **Partial development submodel**
- **Exponent (E) model**
  - **Accounts for diseconomy of scale**
  - **Constant and 3 scale factors**
- **Effort multipliers EM**
  - **15 EMs**

Harmonized COSYSMO 3.0 Interoperability Model

- **Interoperability: The ability of systems to provide services to and accept services from other systems and to use the services so exchanged to enable them to operate effectively together.**
- **Lane & Valerdi [6] propose that interoperability be considered a cost influence in the COSYSMO family**
- **Motivation: if a system is part of a system-of-systems, then that context is reflected in interoperability requirements on the system**
- **Two ways this influence could be manifested are proposed:**
  - **Method 1: Add a new effort multiplier**
  - **Method 2: Adjust the easy/medium/difficult rating scale for system interfaces (part of the Size model)**
- **Both Methods are shown in this presentation; presumably only one would be retained in final COSYSMO 3.0.**

Harmonized COSYSMO 3.0 Size Model

$$Size_{C3} = \sum_{Prods} eReq(Type(Prod), Difficulty(Prod)) \cdot PartialDevFactor(Phase_{Start}(Prod), Phase_{End}(Prod))$$

- **"Prod" is one of the system engineering products that determines size in the COSYSMO family (per [2]). Any product of these types is included:**
  - **System requirement**
  - **System interface**
  - **System algorithm**
  - **Operational scenario**
- **For simplicity in model explanation, each individual Prod is considered separately**
- **There are two submodels:**
  - **Equivalent nominal requirements ("eReq")**
  - **Partial development**

Size Model – eReq Submodel (1/2)

- The eReq submodel is unchanged from [2].
  - Though terminology is a little different
  - Also, see next slide on system interfaces
- The submodel computes the size of a Prod, in units of eReq ("equivalent nominal requirements")
- Each Prod is evaluated as being easy, nominal, or difficult.
- Each Prod is looked up in this size table to get its number of eReq:

Prod Type	Easy	Nominal	Difficult
System Requirement	0.5	1.0	5.0
System Interface	1.1	2.8	6.3
System Algorithm	2.2	4.1	11.5
Operational Scenario	6.2	14.4	30.0

Size Model – eReq Submodel (2/2)

Adjustment for interoperability (Method 2):

- [6] proposes (in its Table 3) that the table that defines the easy/medium/hard rating scale for a system interface (from [2]) be adjusted by adding a new row (the last row in this table):

Easy	Medium	Difficult
Simple messages and protocols	Moderate communication complexity	Complex protocol(s)
Uncoupled	Loosely coupled	Tightly coupled
Strong consensus among stakeholders	Moderate consensus among stakeholders	Low consensus among stakeholders
Well behaved	Predictable behavior	Emergent behavior
Domain or enterprise standards employed	**Functional standards employed**	**Isolated or connected systems with few or no standards**

Size Model – Partial Development Submodel

- **If a project is executing the entire life-cycle, then** *PartialDevFactor* **= 1, so it can be ignored.**
- **This model is not yet mature enough to handle other cases; it will not be considered in today's Delphi**

Harmonized COSYSMO 3.0 Exponent Model (1/2)

- Exponent model is expanded from Peña [4, 9]

$$E_{C3} = E_{COSYSMO} + SF_{ROR} + SF_{PC} + SF_{RV}$$

Where:

- $E_{COSYSMO}$ = 1.06 [2]
- *ROR* = Risk and Opportunity Resolution
- *PC* = Process Capability
- *RV* = Requirements Volatility

Harmonized COSYSMO 3.0 Exponent Model (2/2)

Each of the SF_i ("scale factors"):

- **Is defined by a table of sub-elements, whose ratings are combined to yield a rating for the scale factor**
- **In the most favorable case, has a value of 0**
- **Will be defined in detail during the Delphi**

Harmonized COSYSMO 3.0 Effort Multiplier Model (1/2)

- 15 effort multipliers covered today (summarized on next slide)
  - Detailed definitions during the Delphi
- 13 carried forward from COSYSMO 1.0 [2]
  - Most with few or no modifications
- 1 dropped (Documentation Match to Life Cycle Needs)
- 2 added (CONOPS Understanding, Interoperability)
  - This is Method 1 for Interoperability
- Similarly, Process Capability appears both as an effort multiplier and a scale factor
  - With data calibration, one approach will be dropped

Harmonized COSYSMO 3.0 Effort Multiplier Model (2/2)

- **Here are the 15 effort multipliers:**

Driver Name	Data Item
CONOPS understanding	Subjective assessment of the CONOPS
Requirements understanding	Subjective assessment of the system requirements
Architecture understanding	Subjective assessment of the system architecture
Level of service requirements	Subjective difficulty of satisfying the key performance parameters
Migration complexity	Influence of legacy system (if applicable)
Technology risk	Maturity, readiness, and obsolescence of technology
Interoperability	Degree to which this system has to interoperate with others
# and Diversity of installations/platforms	Sites, installations, operating environment, and diverse platforms
# of Recursive levels in the design	Number of applicable levels of the Work Breakdown Structure
Stakeholder team cohesion	Subjective assessment of all stakeholders
Personnel/team capability	Subjective assessment of the team's intellectual capability
Personnel experience/continuity	Subjective assessment of staff consistency
Process capability	CMMI level or equivalent rating
Multisite coordination	Location of stakeholders and coordination barriers
Tool support	Subjective assessment of SE tools

Bibliography (1/3)

1. "A Generalized Systems Engineering Reuse Framework and its Cost Estimating Relationship", Gan Wang, Garry J Roedler, Mauricio Pena, and Ricardo Valerdi, INCOSE 2014.
2. "The Constructive Systems Engineering Cost Model (COSYSMO)", Ricardo Valerdi (PhD Dissertation), 2005.
3. "Estimating Systems Engineering Reuse with the Constructive Systems Engineering Cost Model (COSYSMO 2.0)", Jared Fortune (PhD Dissertation), 2009.
4. "Quantifying the Impact of Requirements Volatility on Systems Engineering Effort", Mauricio Pena (PhD Dissertation), 2012.
5. "Life Cycle Cost Modeling and Risk Assessment for 21st Century Enterprises", Barry Boehm, Jo Ann Lane, Supannika Koolmanojwong, Richard Turner (presentation), April 29, 2014.
6. "System Interoperability Influence on System of Systems Engineering Effort", Jo Ann Lane, Ricardo Valerdi, CSER 2011.
7. "COSYSMO Extension as a Proxy Systems Cost Estimation" (presentation), Reggie Cole, Garry Roedler, October 23, 2013.

Bibliography (2/3)

8. “COSATMO: Developing Next-Generation Full-Coverage Cost Models” (presentation), Jim Alstad, USC CSSE Annual Research Review, April 29, 2014.
9. "Quantifying the Impact of Requirements Volatility on Systems Engineering Effort” (presentation), Mauricio Peña, Ricardo Valerdi, October 18, 2012 (COCOMO Forum).
10. “Cost Model Extensions to Support Systems Engineering Cost Estimation for Complex Systems and Systems of Systems”, Jo Ann Lane, CSER 2009.
11. “Proposed Modification to COSYSMO Estimating Relationship”, Gan Wang, Ricardo Valerdi, Barry Boehm, Alex Shernoff, INCOSE 2008.
12. “Towards COSYSMO 3.0”, revised PowerPoint presentation by Gan Wang (filename is “COSYSMO 3.0 Definition Outline.Rev.pptx”), May 2015.
13. “COSATMO: Presenting the Harmonized COSYSMO 3.0 Model” (presentation), Jim Alstad, October 22, 2014 (COCOMO Forum).

Bibliography (3/3)

14. “Towards COSYSMO 3.0”, 3rd revision of PowerPoint presentation by Gan Wang (revises [12]) (filename is “COSYSMO 3.0 Definition Outline.Rev3.pptx”), June 2015.
15. “C4ISR Architecture Working Group Final Report - Levels of Information System Interoperability (LISI)”, Department of Defense, Washington DC: OSD(ASD(C3I)) C4ISR AWG, 1998.
16. “The Levels of Conceptual Interoperability Model", Tolk, A., and Muguira, J., Proceedings of the 2003 Fall Simulation Interoperability Workshop. Orlando FL, September 2003.
17. “Towards a Holistic, Total Engineering Cost Model”, Wang, G., Shernoff, A., Turnidge, J., and Valerdi, R., INCOSE Singapore, July 2009.
18. “COSYSMO Reuse Extension”, Wang, G., Valerdi, R., Ankrum, A., Millar, C., and Roedler, G., INCOSE Utrecht, June 2008.

estimate

estimate • analyze • plan • control

Estimating Agile Software Development

Software Cost Estimating for Iterative/ Incremental Development Programs
Agile Cost Estimating

Software and IT Cost IPT
August 2015

SEER by GALORATH

Agenda

- Software versus Hardware estimating
- Fundamental assumptions of current Software Estimating Models
- Iterative and Incremental Development (IID) Programs/Agile Software Development Processes
- Size Metrics
- Software Estimating Processes
- Issues for Program Managers
- Summary

Hardware vs Software Estimating

- We make Software Estimating “seem” to be different, but -
- Both use multiple estimating techniques
- Both use similar techniques:
  - Expert judgment – utilize SME inputs for a task
  - Analogy – estimate based on past examples
  - Parametric – mathematical relationships
- Top down – estimate total cost and allocate
- Bottom up – estimate each detailed tasks/CSCIs

Most estimates use a blend of methods

Fundamental Model Assumptions

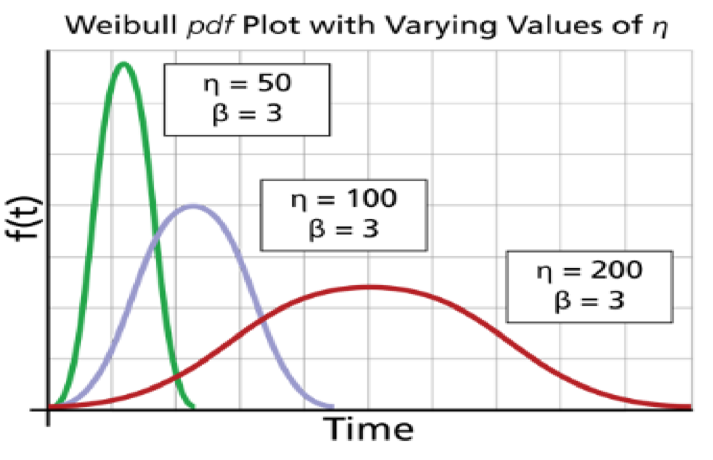
Fundamental assumptions of most Software Estimating Models

- There is a fixed relationship between size and effort, e.g.

  (Effort**n)*Time = Size/Technology

- Results are then modified by current trends and analyses

- Total effort can be distributed by a mathematical model; e.g. Weibull, Rayleigh

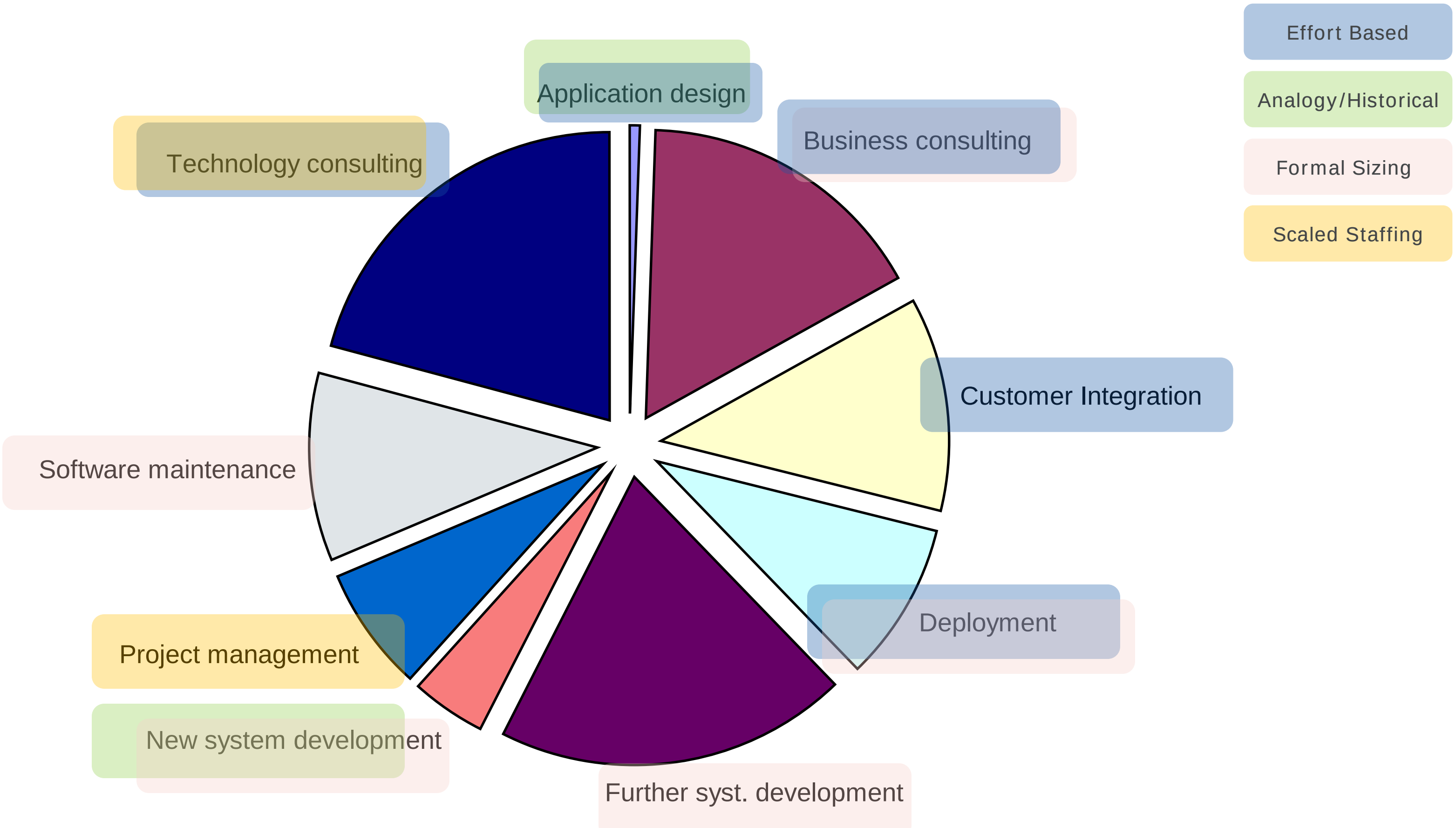


Software Development

- While there are many approaches to Software Development, they can generally be placed into 2 categories:
  - Plan Driven – following a version of the Waterfall Development Process
  - Iterative Driven – following a "version" of the Agile Development Process
- Plan Drive programs have an assumption of some reliable/realistic size metric, for example:
  - Source Lines of Code (SLOC)
  - Function Points
  - Use Cases, User Stories, Web Pages

Sizing Approaches Observed in Industry

Not all technology and product types can use the same sizing methods



Software Development

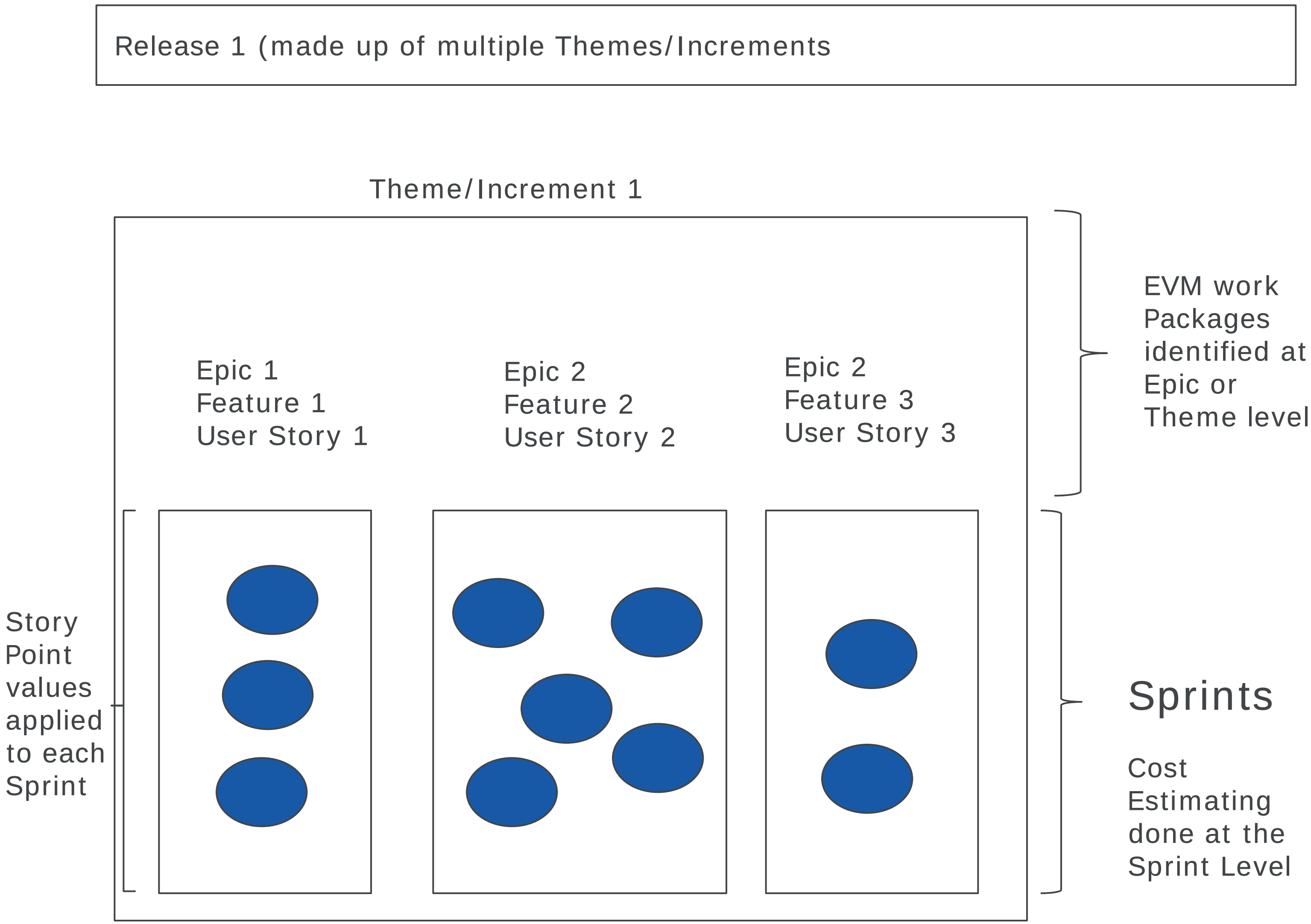
- Iterative Drive programs, by nature, start with a less well-defined metric
  - Therefore, they may require alternative estimating approaches
- This briefing will focus on the challenges of estimating an iterative program using Agile software development
- In practical experience the terms iterative, incremental and agile may be used interchangeably
- Look for terms like features, epics, time-boxes, releases

While Incremental/Agile programs say they do not have development metrics, I have almost always found them in the development room

IID Programs' Key Terms

- **IID** is an approach to building software in which the overall lifecycle is composed of iterations or sprints in sequence
  - Each Iteration is a self-contained mini project
  - It grew out of the increased application of Agile Development techniques
- In many Federal programs, **increments** are 6 -12 months in length and each increment is composed of multiple **iterations/sprints** of 1-6 weeks [1]
- Sprints can be combined into increments, releases, epics, themes; however, the Sprint is the key Work unit
- Time-boxing is the practice of fixing the iteration or increment dates and not allowing it to change
- This approach is gaining favor in large federal programs

Agile Building Blocks*

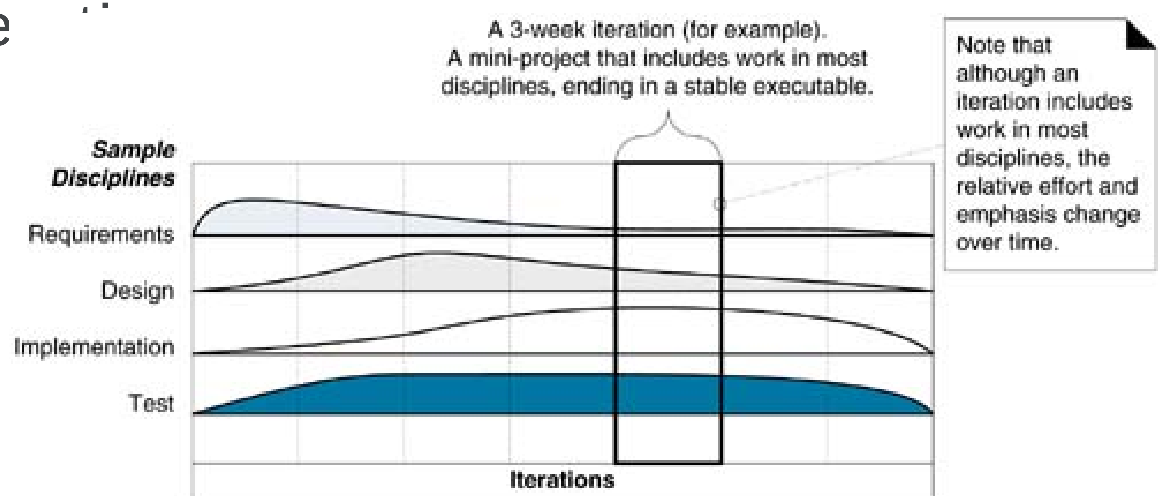


* These "building blocks" are program specific and may be called by different names

Each Iteration/Sprint is a Mini Project (in theory)

SEER by GALORATH

- Each iteration/sprint includes production-quality programming, not just, for example, requirements analysis
  - The software resulting from each iteration/sprint is not a prototype or proof of concept, but a subset of the final system
- More broadly, viewing an iteration as a [1] self-contained mini project, activities in many disciplines (requirements analysis, testing, etc.) occur within a [2] single ite



IID

- Although IID is in the ascendency today, it is not a new idea
  - 1950s “stage-wise Model” – US Air Defense SAGE Project
  - IBM created the IID method of Integration Engineering in the 1970s
- IID Programs tend to be less structured in the beginning, and therefore reliable estimates of cost and schedule may not be available until 10-20% of the project is complete[4]

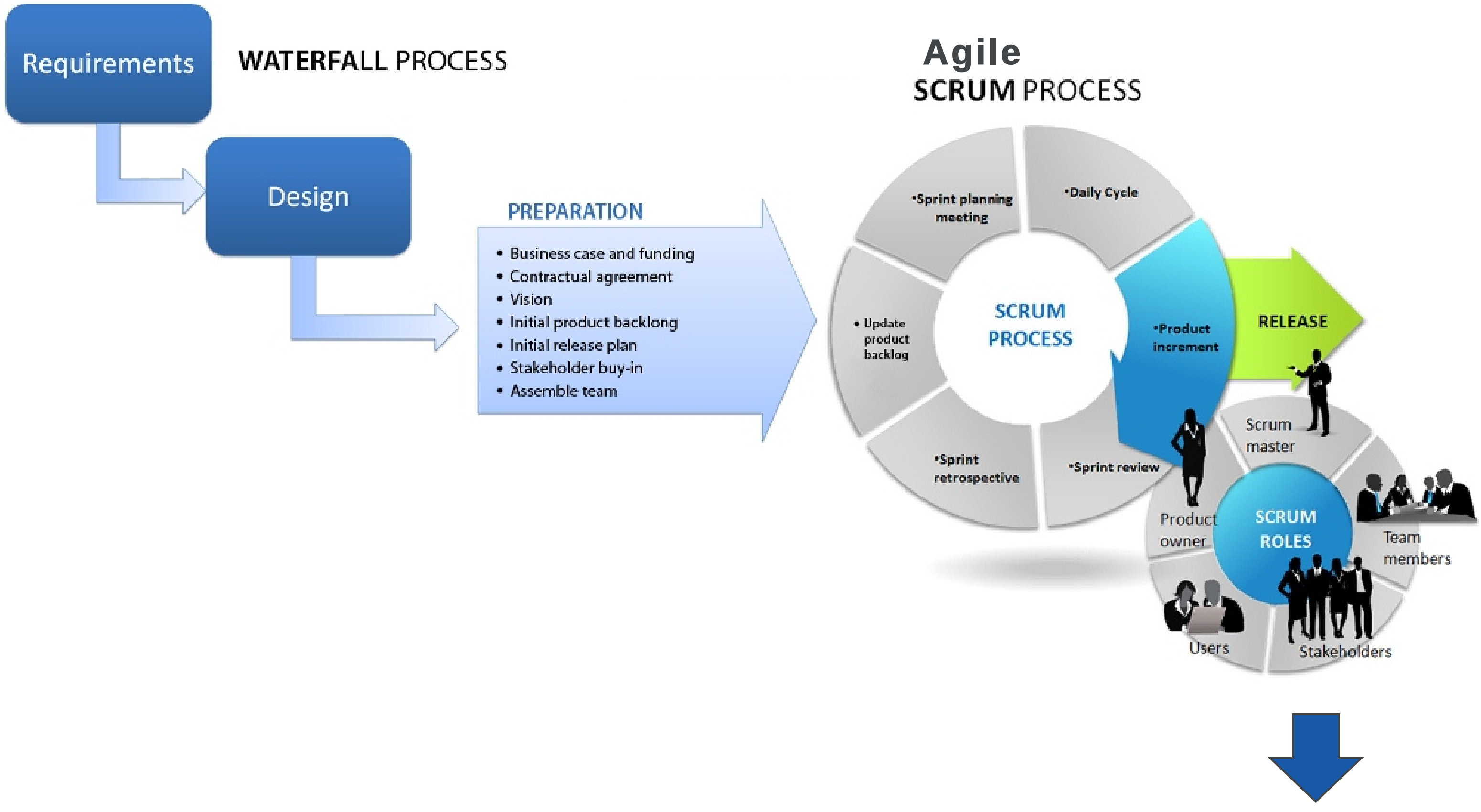
(in a recent program I saw a cost variance during the first 4 increments of 45% per size metric)

- The current emphasis on agile software development processes maps directly into the IID Concept

What is Agile Software Development?

- In the late 1990s, several methodologies received increasing public attention
- Each had a different combination of old, new, and transmuted old ideas, but they all emphasized:
  - Close collaboration between the programmer and business experts
  - Face-to-face communication (as more efficient than written documentation)
  - Frequent delivery of new deployable business value
  - Tight, self-organizing teams
  - And ways to craft the code and the team such that the inevitable requirements churn was not a crisis

Hybrid Agile Development/Acquisition



Testing and Sustainment ?

How Formal Is Agile?

Manifesto for Agile Software Development

"We are uncovering better ways of developing software by doing it and helping others do it.

Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more."

Agile is NOT a Method – it's a mindset!
Individual Methods are Formal – sort-of

Why Does "Old School" Fear Agile?

- Remember these words from the Manifesto?
  - *Individuals and interactions over processes and tools*
  - *Working software over comprehensive documentation*
  - *Customer collaboration over contract negotiation*
  - *Responding to change over following a plan*

- *How can a manager possibly control this?*
- *How can it be measured?*
- *How can we allow the developers to talk to the customer?*
- *What? NO DOCUMENTATION?*

For Management: Agile equates to Loss of Control

Common Myths about Agile

Myth	Reality
Silver bullet / magic	Actually very hard work!
Has no planning / documentation / architecture	Just the minimum possible
Is undisciplined or a license to hack	Disciplined, business driven work
Is new and unproven / just a fad / not being used by industry leaders	Not anymore. Many large and small organizations using it
Only good for small projects	Also used successfully on medium and large projects

Differences of Agile and Non-Agile

Agile	Non-agile
Prioritize by value	Prioritize by *dependency*
Self-organizing teams	*Managed* resources the minimum possible
Team focus	*Project* focus
Evolving requirements	*Frozen* requirements
Change is natural	Change is *risky*

- Recent observations regarding the utilization of Agile development approaches within the Federal Government:
  - May work best when the project is more requirements-driven than schedule-driven
  - Beginning to see common usage in Department of Defense (DoD) unclassified (e.g. Marine Corps) and classified programs (e.g. Naval Reconnaissance Office [NRO])

Some Agile Definitions

- Scrum – one of many Agile methods used for development
  - Uses a time box approach with a fixed team and key members
- Sprint – a time box used in Scrum. Usually between two to four weeks. Sprint starts when planning is done and ends at a specific point in time
  - *Waterfall = Run this distance, Agile = Run for this long*
- Sprints are combined into logical groups called Epics, Themes, Releases, Increments
- Scrum Team
  - Product Owner – Person responsible for success of project. Must have direct stakeholder/customer contact
  - Scrum Master – Process facilitator and coach. Is not a developer or similar team member
  - The Team – Self organizing collection of cross-functional specialties that perform the work

Welcome to Agile

- What is an agile development approach?
- Depends on the *flavor*:
  - Agile Modeling
  - Lean Development (LD)
  - Adaptive Software Development (ASD)
  - Exia Process (ExP)
  - Scrum
  - eXtreme Programming (XP)
  - Crystal methods
  - Evolutionary – EVO
  - Feature Driven Development (FDD)
  - Dynamic Systems Development Method (DSDM)
  - Various Unified Processes (UP): agile, essential, open
  - Velocity tracking, and more!

What do they have in common?

- Agile projects are focused on key business values
  - What does the client really, really, *really* want?
  - Deliver what the client wants at the end of the project, not what the client wanted at the beginning of the project
- They all contain a project initiation stage (aka planning)
  - Project scope, constraints, objectives, risks are all officially documented
- Short (very short) development of chunks of features/stores/requirements/needs/desires (aka sprints)
- Constant feedback
  - The one place where we can actually find short meetings
- Customer participation is MANDATORY or no-go!
- Refactoring; as in, do it again and this time get it right, or better

Other Current Research

Empirical evidence indicates development costs may be reduced by 10 to 20 percent for Iterative Driven Programs. In a "The Raytheon Agile Journey" a presentation by Cindy Molin (Director, SW Engineering) and Katherine (K) Sementilli (Deputy, SW Engineering), Raytheon Missile Systems on June 22, 2012 the following efficiencies based on agile development are observed (based on over 250 projects and over 5 million ELOCs):

Agile Development Results

- 20% of Raytheon SW Engineering Development Productivity
- 25% productivity increase Agile vs Non-Agile
- 10% variability reduction Agile vs Non-Agile
- 50% faster for Agile vs Non-Agile
- Time on task for an average work day 30% more for Agile vs Non-Agile

Scrums and Sprints

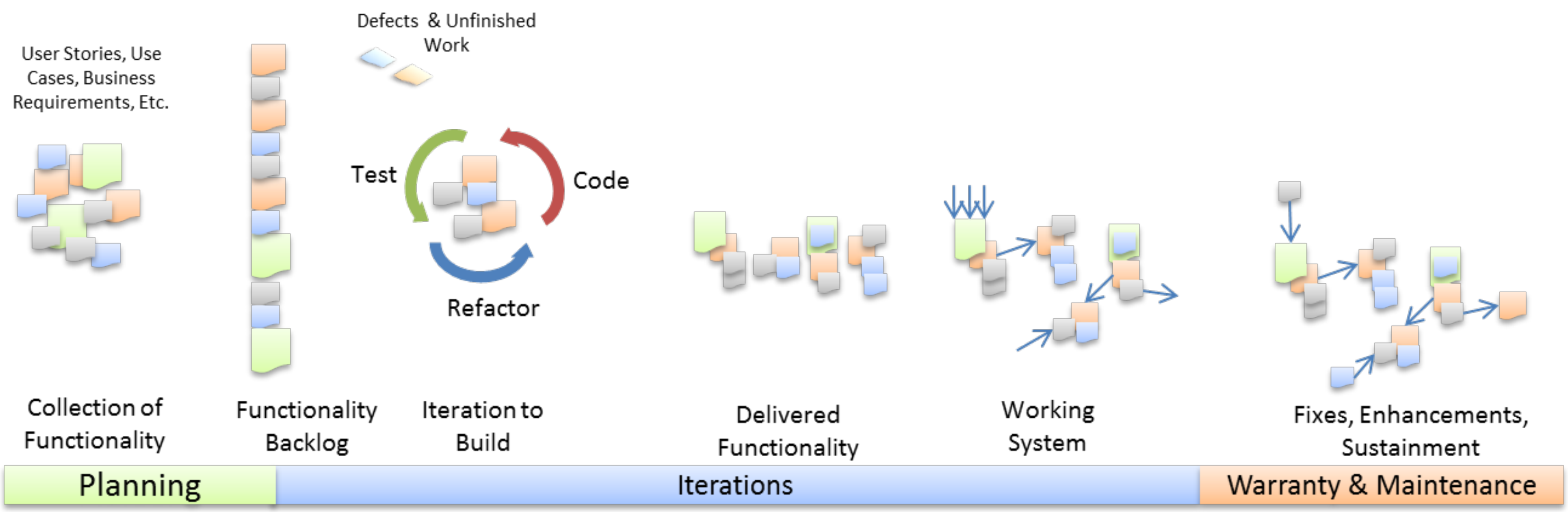
SEER by GALORATH

- Scrum Size:
  - 1-10 people (have seen up to **20**)
- Sprint Length:
  - 1-6 weeks (have seen up to **13** weeks) *(13 conveniently give 4 sprints per year)*
- Story Points* per Sprint:
  - 6-9 Story Points per Sprint
- There seems to be a real avoidance of using Function Points or SLOC in many of these efforts.
- (*But trust me a size metric exists somewhere within the development community*)

* I have Use Case, Feature Point, and other metrics for specific agile development programs, but I am not sure they are transferable

The Agile "Life Cycle"
(Scrum Example)

- Focus is on what features can be delivered per Sprint/Epic/...
- Defined what functionality will be delivered at the end?



Things To Consider When Estimating Agile Projects

- Gauge the Organization:
  - Assess teams interactivity and motivation
  - Teams familiarity with process
  - What is the real role of QA
  - Do they really have everyone in place
  - Is the Product Owner really a PO?

- Revisit the Estimate After One or Two Iterations
  - Repeat the first bullet!

- Pay Attention To Backlog
  - Could Indicate Process Immaturity…
  - …but Is More Likely Volatility or scope creep

Agile Labor Categories

- Most Agile methodologies have only a few defined categories.
  - Scrum only has: Product owner, Team, Scrum Master

- However, the work of each category is still being performed
  - Analysis, requirements, and design done by team
  - Coding and testing done by the team
  - CM is performed by team
  - QA remains a PM reporting activity

- Need to have a Product Owner and expert (SME)
  - The SME is an integral part of the team
  - Sometimes acts as Product Owner
  - Must have contact with the Stakeholders
  - Responsible for success of the project (accountable)

Size Metrics

- Very few software development projects start without some concept/metric of how long and how much time the development will take
- The Estimator's challenge is to "quiz" that out of the metric
- I have always felt more confident in my answer when the estimation team's size metric/proxy and the development team's effort proxy were the same
- I believe there is more estimation error from poor sizing than from estimation models or processes
- Common size metrics – a “can” be valid

SLOC	Function Points
Use Case	Stories

- Agile Software Development seem ot often avoid SLOC and Function Point metrics

The Rise of Story Points

A "User Story" is a simple statement about what a user wants to do with a feature and the value the user will gain from that feature.

- The User
- What the user wants to do with a feature
- Value gained by the user
- Acceptance Criteria
- Front and back of a card

Story Points represent a value given to a user story that is used to measure the effort required to implement the story

It is a number that represents story size based on how hard/how complex a story is to implement.

User Story Defined

- A User Story is a simple statement about what a user wants to do with a feature and the value the user will gain.
- Consider a User Story as a thin vertical slice through the system.
- User stories are written from the user perspective in a way that can be easily understood.
  - Technical jargon is avoided.
- Acceptance Criteria is usually written at the same time.
- The Project Owner is responsible for the user stories
  - Written by the Sponsor
  - Reviewed by the project team
- Usually written on cards
  - Story on the front
  - Acceptance Criteria on the back

Story Points Defined

- Story Points represent a value given to a user story that is used to measure the effort required to implement the story.
  - Points are assigned to User Stories
  - Those point values are later used to estimate
- It is a number that represents story size based on how hard a story is to implement.
- When assigning points to a story focus is placed on effort to implement but not on time. That comes later during estimation.
- Many would say that a story point is an arbitrary measurement that depends on the team.
  - There is some truth to that statement
  - We can manage that to some extent

Things To Consider When Counting SPs

- Number of interfaces with the outside world
  - Most agile teams do not function in a vacuum and must consider the needs of the rest of the organization.
  - Certain tasks and artifacts may be required by the organization or by governance that the team will have to support.
  - This could vary depending on the story or the product.
- Complexity of the Code
- Number of required tasks
  - Depending on the story certain formal or informal tasks may not be necessary.
  - Regulation can play into this
    - There may be more checks and balances to a story due to governance
- Coordinating people takes effort too

Some Drawbacks To Story Points

- Story Points are team dependent!
  - Members of different teams will have different levels of experience leading to different perspectives related to how hard a story is.
- Points don't easily scale across different projects.
  - How one team's points can vary.
- Inflation can occur as soon as the second sprint
  - Teams often blame not delivering the Story due to a faulty count. That is usually not the reason!
  - As a consequence a natural inflation appears during the next count.

Steps In Pointing Stories

- Start with a point system that everyone agrees on.
  - It is best that the system used to compare against is used across projects just for consistency.
- Identify at least one base story.
  - It could be a medium or average size story but it doesn't have to be.
  - This base story will be used to compare other stories against.
- Review each story as a team.
  - Discuss its complexity and size considering what it will take to develop the story.
  - Compare the story against the base story and agree on a score for the story selecting a number in the sequence being used.
- The team must agree unanimously on a score
  - You can use games to determine score.

Story Point Velocity

- Velocity measures how much of something can be achieved over a fixed period of time; e.g. how many Story Points are completed during a Sprint
  - You could do this at the User Story level but you have no relative measure between User Stories
- Velocity is a "team" measurement – not the individual
- Iteration Duration / Completed Total SP = Velocity
- Iterations needed = Total SP / Velocity
- Don't change the duration and use the same result

Quiz: A project has a total size of 365 Story Points. The team has a velocity of 25/SP per iteration

How many iterations will the project take?

A Process For Assigning Points

- Review a group of stories as a teams
  - Discuss the complexity of the story and the activities that will be needed to implement the story.
  - Consider unknown technology and new processes required.
  - Each team member selects a number from their Planning Poker deck of cards
  - The team discusses until they agree on a point value.
  - If the team can't agree, the story is sent back to the Product Owner to be rewritten.
  - If a story point cannot be agreed upon the problem usually resides in the story itself.

Some Drawbacks To Story Points

- Story Points are team dependent!
  - Members of different teams will have different levels of experience leading to different perspectives related to how hard a story is.
- Points don’t easily scale across different projects.
  - How one team’s points can vary.
- "Inflation" can occur as soon as the second sprint
  - Teams often blame not delivering the Story due to a faulty count. That is usually not the reason!
  - As a consequence a natural inflation appears during the next count.

Disadvantages

- Difficult to explain
  - May resort to comparing Poodles to Great Danes
- Teams have different interpretations when working independently
  - Story Points need to be scored as a team
- Tendency to skip over detailed iteration planning by assuming a velocity * SP = work
  - Still need to break the User Story into tasks and estimate the capacity for the sprint
- Takes a while to trust the results
  - But this is common for all new sizing measures!

Advantages To Story Points (a few)

- Prevents Managers from converting a SP into a Calendar day (They have to know the velocity)
- Promotes cross-functional behavior (teams can compare similar things)
- Points do not decay (when compared to ideal days)
- They are a “Pure” size measure relative to other known things
- People are pretty good at generating a valid relationship of size



Four Estimating Processes

SEER by GALORATH

- **Process 1: Simple Build-up approach** based on averages can be defined as:
  - Sprint Team Size (SS) x Sprint length (Sp time) x Number of Sprints (# Sprints)
- **Process 2: Structured approach** based on established "velocity" – most often used internally by the developer since detailed/sensitive data are available to them
- **Process 3: Automated Models approach** based on a size metric – which may be difficult to quantify
- **Process 4: Factor/Complexity approach** based on data generated in early iterations

A Word About 2014 Rates

- Developers and Tester - $70 to $200 per hour, median team rate about $125
- Agile Coach - $100 to $200 per hour, average about $150
- Business Analyst - $125
- Average Team Rate of about $115

WARNING: THESE ARE BROAD AVERAGE I HAVE FOUND THIS YEAR

Process 1: Build-Up Approach

When a program is comprised completely of agile sprints, we can use industry norms or program plans to develop an estimate

- Process 1 is defined as:
  - SS x Sp time x # Sprints
    - SS (normally 1-10 people) x Sp time (normally 0.25 to 1.25 months) x # Sprints
    - Frequently used by independent estimators since actual data are often unavailable
    - Remember to factor in time for demonstrations/user feedback
    - Can develop a point estimate and a range
    - Works well for small programs

The weakness of this approach is justifying the team size, number of sprints, sprint length and total required to meet the requirement

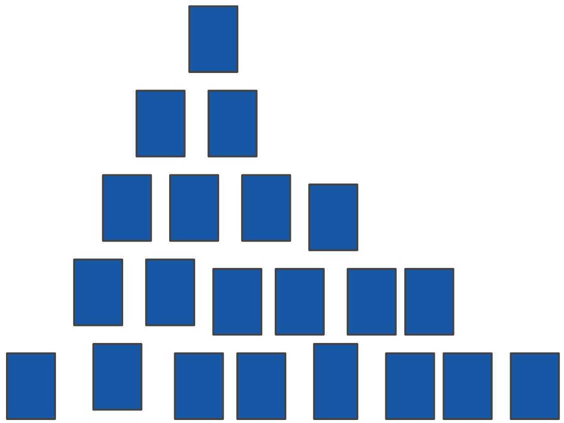
Process 2: Structured Approach based on “Velocity”

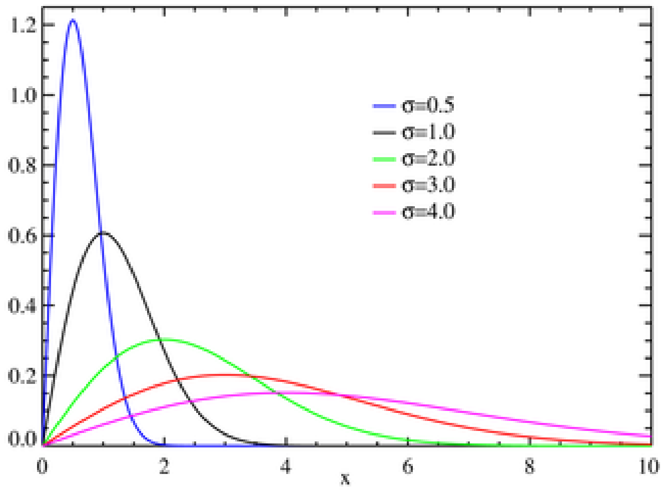
- Process 2 can be summarized by:
  1. Express requirements in the same size metric used by the developer; normally Features, Feature Points, Use Case Points, Story Points, … What the size metric is unimportant as long as it is consistently used across this program*
  2. (optional). Use a process to rank the size metric: small, medium, large using something like Fibonacci sequence, planning poker
  3. Estimate and/or document the velocity (number of size metrics per time period) at which the Agile team has worked
  4. Estimate and/or document the historic cost per size metric for the Agile team
  5. Spread the sprints over time to develop time-phased estimate

* I would hope that over time we could develop standards for agile development across the various size metrics and programs. However, since these metric often do not conform to a “standard” this is an elusive task. But an average over several early interactions may be very accurate for a specific [program.

Moving to Automated Models

- Step 5 of the previous slide suggested you time-phase the Sprints
  - When you do this, the results often resemble the Rayleigh Function used in modern software models





- This observation leads to the third estimating process

Process 3: Automated Model Approach

- The “Parameter” settings within automated models can be adjusted to estimate costs and schedule for complex/large projects
  - The “environmental factors” in SEER, PRICE, SLM, and COCOMO II have been adjusted to reflect Agile practices and therefore Iterative Development
  - Remember, the size metric is still the key cost driver, which is even less certain in agile programs than traditional ones

Process 4: Factor/Complexity Approach

- In a normal IID program, the initial program estimate must be based on broad parameters with wide ranges – analogy to previous programs and/or generic models
- Specific iterations/sprints can be estimated using the agile estimating processes previously presented
- The real question is: how do we estimate the cost of future Increments (time boxes)?
- The following slides present Process 4 Factor/Complexity Approach

Process 4: Factor/Complexity Approach

- Step 1: Select a Baseline Increment (often the last successful increment) for the program
- Step 2: Carefully analyze this baseline increment – this analysis could be based on SLOC, function points, features, requirements, dollars, or some other metric
- Step 3: For each new increment, compare the expected functionality and complexity of the new increment to the baseline (or last successful) increment

  - Notional functional and complexity factors are presented on the next slide

Process 4: Factor/Complexity Approach

Scale	Functional Description	Effort Multipliers
- - -	Significantly less functionality to be delivered	0.5
- -	Moderately less functionality to be delivered	0.7
-	Slightly less functionality to be delivered	0.9
=	Functionality equivalent to Increment X	1.0
+	Slightly more functionality to be delivered	1.3
+ +	Moderately more functionality to be delivered	1.7
+ + +	Significantly more functionality to be delivered	2.0

Scale	Complexity Description	Effort Multipliers
- -	Significantly less complex	0.7
-	Slightly less complex	0.9
=	Complexity equivalent to Increment X	1.0
+	Slightly more complex	1.3
+ +	Significantly more complex	1.7

- These initial set of factors came from the environmental factor from traditional software cost models
- Step 4: Because each Increment is a mini project, use a Rayleigh or simple Beta Curve (such as a 60/50 Beta curve) to phase costs
- However, do not be surprised if you encounter programs that are truly operated and manages as Level of Effort (LOE)

Issues for Project Management

- Cost and Schedule modelers usually want well-defined program requirements and size metrics early in the lifecycle – the nature of IID programs argues against this
  - IID programs tend to be less structured in the beginning, and therefore reliable estimates of cost and schedule may not be available until 10-20% of the project is complete

11

- Initial contracts tend to be Fixed Price or LOE
  - This does not imply poor value to the project
  - It does imply that key "value-added" metrics may not be identified or collected
- "Time Boxing" tends to resolve the individual scheduling issues, but not the total program length issue
  - A specific cost estimating strategy is required to accurately plan for resources

Issues for Project Management

- If a program has too many planned Increments (10 or more), it may not be a well-defined program and could spin out of control or just become an LOE research project
- Establishing and monitoring metrics becomes critical
- “To be able to adopt an empirical approach to project management and control, we must be able to objectively demonstrate and measure how much progress the project has made in each iteration
  - Possible ways to measure progress include:
    - Number of products and documents produced
    - Number of lines of code produced
    - Number of activities completed
    - Amount of budget/schedule consumed
    - Number of requirements verified to have been verified implemented correctly”

Schedule Analysis

- Due to the short length of increments (generally 9-12 months) and continuity between increments, phasing the costs within a specific increment is less important
- However, the “million dollar questions” for incremental and agile programs (where requirements definition and documentation are less detailed, and the development is more flexible/emergent) are:
  - What will the program look like at Initial Operational Capability (IOC)?
  - How many increments will it take?
  - How long is each increment going to last?
- Cost estimators are going to have to adjust, and examine these programs as a schedule analyst might to produce credible lifecycle estimates

Summary

- Fixed Price and/or LOE contracts in the early phases should be written so that key "value-added" metrics are collected and reported during each increment
- Estimators may have to employ a variety of software estimating methodologies within a single estimate to model the blended development approaches being utilized in today's development environments
  - An agile estimating process can be applied to each iteration/sprint
  - Future Increments can be estimated based on most recent/successful IID performance
- Cost estimators will have to scrutinize these programs like a schedule analyst might to determine the most likely IOC capabilities and associated date
  - The number of increments are an important cost driver as well as an influential factor in uncertainty/risk modeling

Summary

- All of the estimation methods are susceptible to error, and require accurate historical data to be useful within the context of the organization
- When developers and estimators use the same “proxy” for effort, there is more confidence in the estimate

Recommended Reading

- “The Death of Agile” blog
- “Agile Hippies and The Death of the Iteration” blog
- Story Point Inflation

Endnotes

- *1, 2, 4, 10, 11:* Larman, C. (2010). *Agile and Iterative Development: A Manager's Guide.*
- *3:* Kilgore, J. (2012). Senior Associate, Kalman & Company, Inc.
- *5, 6, 7, 8:* Agile Alliance. (2012). *Agile Alliance*. Retrieved 2012, from http://www.agilealliance.org
- *9:* Coaching, T. L. (n.d.). Rally Software Scaling Software Agility.
- *12:* Bittner, K., & Spence, I. (2006). *Managing Iterative Software Development Projects.* Addison-Wesley Professional.

Additional References

- Cohn, M. (2009). *Succeeding with Agile Software Development using Scrum.*
- Dooley, J. (2011). *Software Development and Professional Practice.*
- Gack, G. (2010). *Managing the Black Hole.*
- George, J., & Rodger, J. (2010). *Smart Data (Enterprise Performance Optimization Strategy).*
- Royce, W., Bittner, K., & Perrow, M. (2009). *The Economics of Iterative Software Development: Steering Towards Better Business Results.* Addision Wesley Professional.
- Smith, G., & Sidky, A. (2009). *Becoming Agile in an Imperfect World.*

Contact Information

- Bob Hunt
  - Email: Bob.Hunt@GalorathFed.com
  - Phone: 703.201.0651

How Much Does Software Maintenance Cost?

Deputy Assistant Secretary of the Army for Cost and Economics

11 August 2015

U.S. Army Software Maintenance Cost Estimation Initiative

Provide the Department of the Army with the ability to accurately estimate, budget, allocate, and justify the software maintenance resources required to meet evolving mission and service affordability requirements across the system life-cycle

Critical Software Maintenance Questions

- How much funding does each Army system need to maintain its required mission capability? (RDTE, Production, OCO, PPSS, etc.)
- How many dollars were allocated to a given system from all sources to upgrade and maintain the capability embedded in the software?
- How were those dollars programmed and executed? What did the Army actually buy?
- What was the mission impact of enterprise/system SWM investments?
- Were Army SWM investments optimized over the total systems software portfolio?
- How much funding is required to preserve a viable maintenance capability for a given system/organization/domain?
- How much funding is required to: 1) keep the system operational at an acceptable capability level? 2) prevent catastrophic events?
- How do “in-process” requirements changes impact maintenance cost and product output?

Software System Size Growth

107 - AH-64As

1620 - AH-64Ds

Apache Software Growth
300 KSLOC to Over 1.4 Million SLOC
Since 1984

Software System Configuration Complexity

4,300 - M1A1 & variants
580 - M1A2 & variants
580 - M1A2 SEP & variants

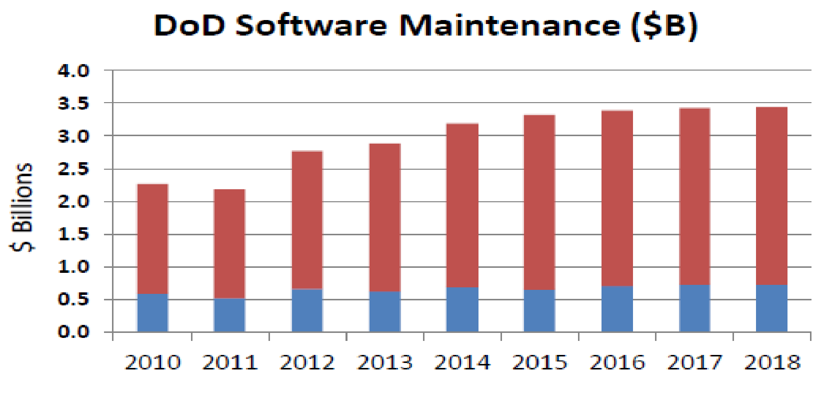
- Complex system interfaces
- Multiple software change drivers
  - End user requirements
  - Mission evolution
  - System interoperability
  - Change mandates
  - Security requirements
  - Technology updates
  - Technical debt

Army Software Engineering Center Requirements Growth

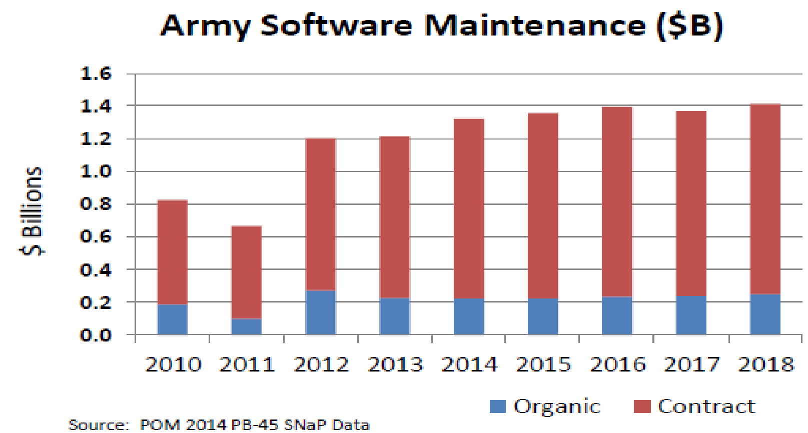
	STARTING DATA POINT		ENDING DATA POINT		WORKLOAD
	YR	#	YR	#	
Number of Systems/Programs	1983	37	2011	400	+1081% over 28 years
Software Releases	1997	64	2011	378	+591% over 14 years
Software Licenses	**2004**	**34,205**	2011	131,037	+383% over 7 years
System Size/Complexity	1970	Small	2011	Med-Large	500+%
PPSS Requirements	**2003**	**$126M**	2011	**$569.5M**	+452% over 8 years
Customer Requirements	2001	$179M	2011	$841.5M	+470% over 11 years

Source: http://www.sae.org/events/dod/presentations/2012/dod_maintenance_symposium.pdf

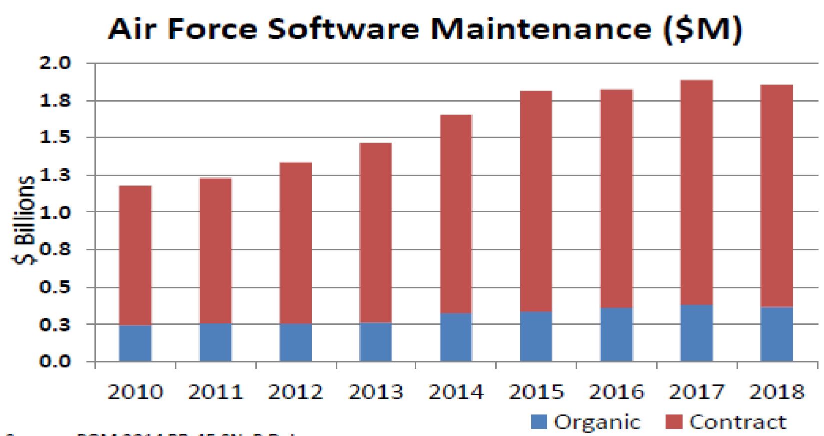
DOD SW Maintenance Funding (Estimates)



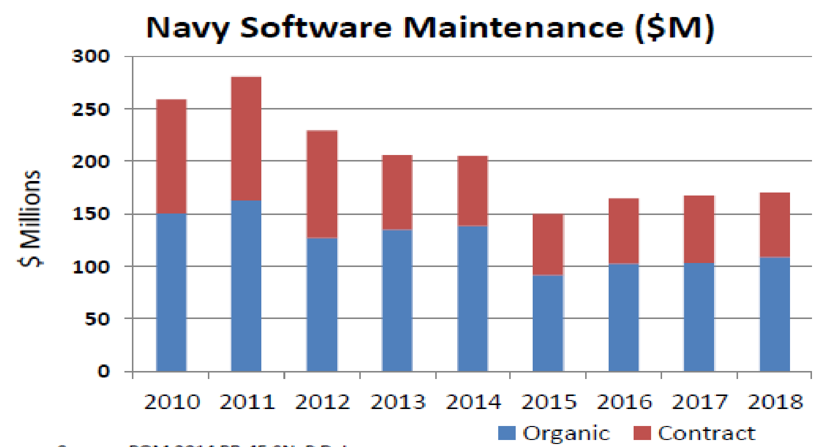
Source: POM 2014 PB-45 SNaP Data



Source: POM 2014 PB-45 SNaP Data



Source: POM 2014 PB-45 SNaP Data



Source: POM 2014 PB-45 SNaP Data

Do More, Without More

Frank Kendall
Under Secretary of Defense for AT&L

- Achieve Affordable Programs
- Achieve Dominant Capabilities While Controlling Life Cycle Costs
- Incentivize Productivity in Industry and Government

Cause and Effects

Significant Overseas Contingency Operations (OCO) dollars available to fund Army SWM efforts for the past 10-15 years

SWM cost efforts focused on high-level planning numbers for requesting funding

Lack of software maintenance actual cost tied to execution output visibility

Inability to effectively estimate software maintenance costs

Key Cost Related Issues

1. Discordant SWM maintenance definitions and cost accounting accrual structures (system, functional, organizational, etc.)
2. Non-aligned cost, resource, and software technical SWM information / systems
3. Volatile change requirements - execution priorities
4. Multiple funding streams (separately managed)
5. Minimal reported contractor performance data (cost/schedule/product output)
6. LOE management structures - LOE resourcing
7. Immature organizational SWM business and technical processes
8. Minimal enterprise level SWM governance/policy (DOD, Army, etc.) - low level cost management autonomy

Software Maintenance Integrated Cost Estimation Methodology

Approach

- All major Army software maintenance organizations were engaged to understand what people do and when they do it
- Cost and technical data was collected from a sample set of programs
- An initial estimation model/methodology was developed and validated based on this data
- Supporting constructs included a tailorable SWM WBS, a relevant set of software functional domains, and a refined set of data requirements
- This model was successfully applied on a set of Army and Air Force pilots, in parallel with the current estimation methodology

Software Maintenance

Software Maintenance

- All activities associated with modifying a software product/system after delivery

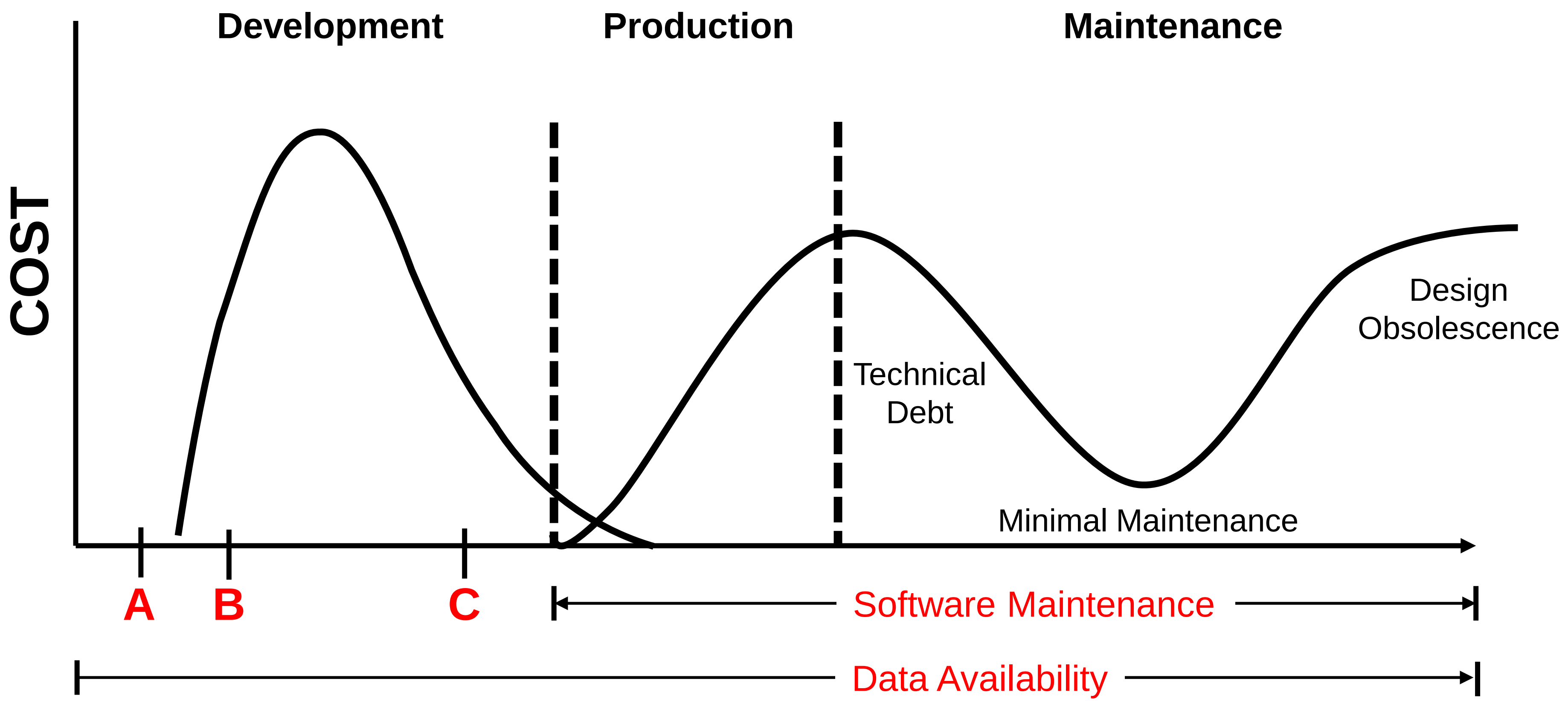
Software Maintenance Costs

- May be directly allocated to a single system or treated as “shared” organizational costs
- Costs are aggregates of outputs/activities executed under multiple funding sources
- Includes software enhancements (RDTE, OCO, Production) and software corrections/adaptations/etc. (PPSS, OMA)
- Costs not aligned with software maintenance output products/activities

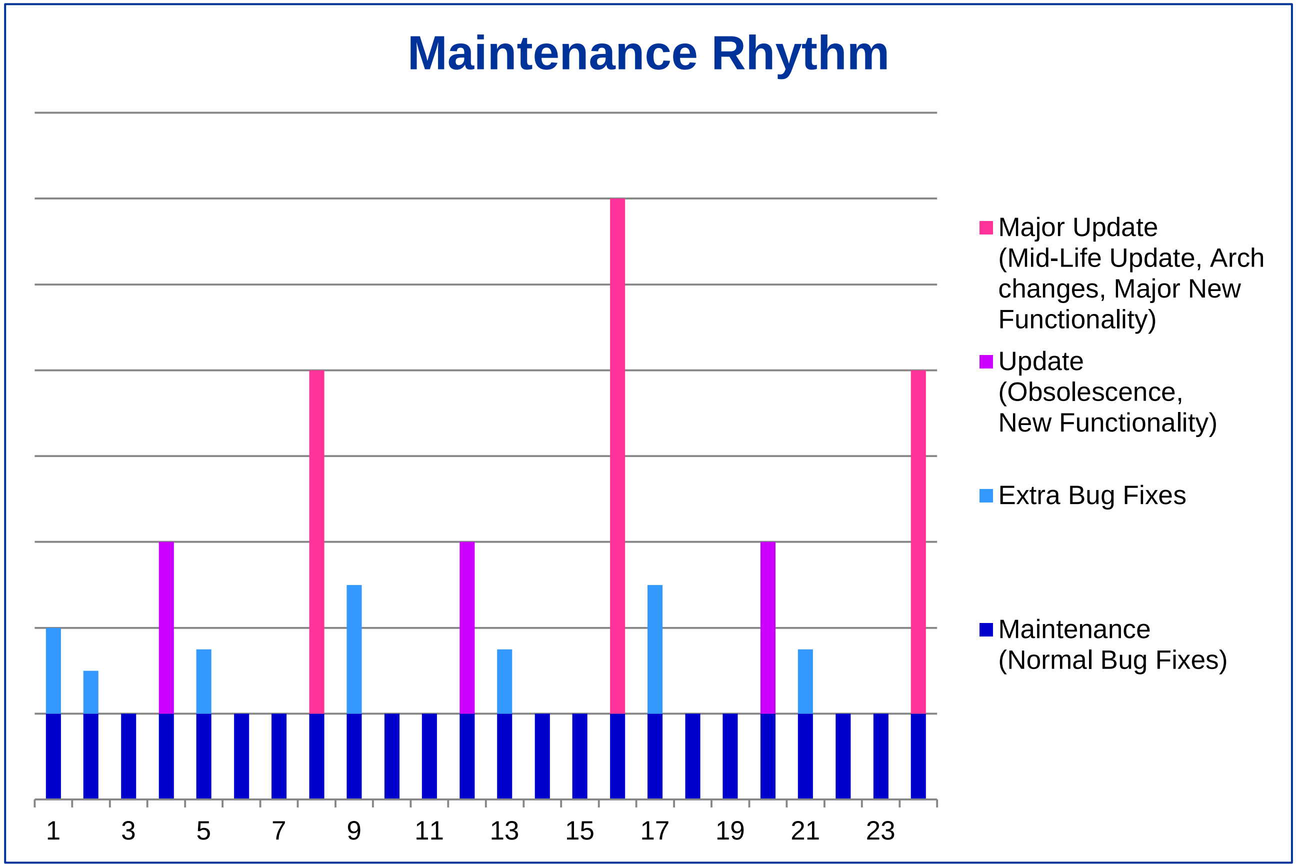
Software Maintenance Cost Estimation Requirements

- Need to effectively estimate and justify software maintenance costs across the system lifecycle
- Estimates required at all phases of a program: beginning before milestone A and continuing through O&M
- Current methods are inadequate and do not provide the information needed by decision makers
- SWM costs are currently estimated as a percentage of the development costs

Notional Software Maintenance Life-Cycle Cost Model

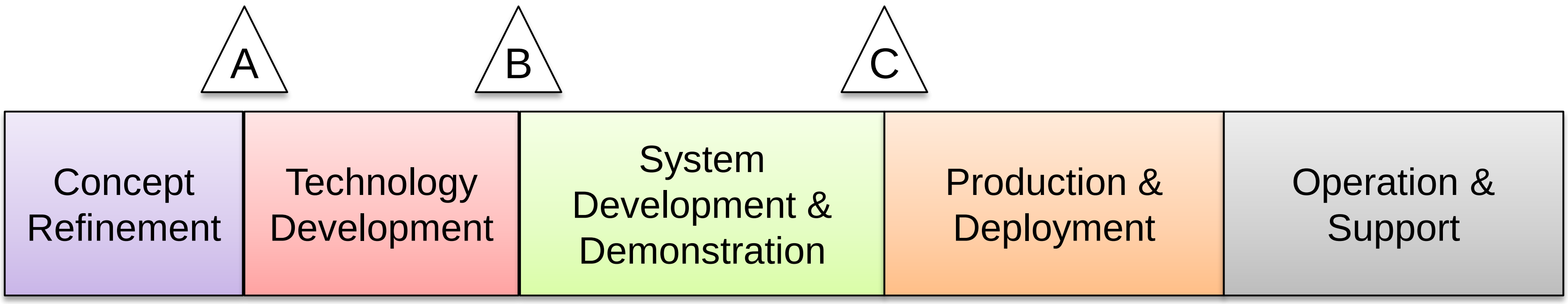


Software Maintenance Release Profile



Cycles are different for different programs
User needs drive release content

DOD Acquisition Lifecycle Model



- Programs may be in any lifecycle phase
- Estimates are required at the major milestones and periodically after milestone C
- Estimation considerations:
  - Availability and quality of program data
  - Different CERs at different estimation points and for different system characteristics (e.g. domains)
  - Different types of CERs: parametric, ratios, trends

Software Maintenance WBS

Software Maintenance

System Specific

1.0 Software Change Product

- Change requirements
- Change development
- B/L Integration & Test

System Specific

2.0 Project Management

- Planning
- Execution management
- Configuration management
- Resource & team management
- Contracting management
- Measurement - reporting

System/Non-System Specific

3.0 Software Licenses

- License - Right to Use
- License - Maintenance
  - COTS
  - NDI
  - Other

System Specific

4.0 Certification & Accreditation

- Security
- Safety
- Networthiness
- Airworthiness

System/Non-System Specific

5.0 System Facilities

- Hardware
  - Software development assets/workstations
  - System integration & test facilities
  - Test equipment - tools
- Facility Operations

System Specific

6.0 Sustaining Engineering

- Engineering Support
  - Test Support
  - Software Delivery
  - Technical Studies
- User Support

System Specific

7.0 Field Software Eng.

- On-Site technical assistance
- Problem Troubleshooting
- S/W Installation
- Operational Assistance
- On-site Training

Non-System Specific

8.0 Operational Management

- Operations
- Organization management
- Personnel management
- Financial management
- Information management
- Process management
- Change management

Version 4.4c

Software Maintenance WBS

- Common structure that includes all potential software maintenance products and activities - “what’s in” - “what’s out”
- Defines the superset of program software maintenance cost elements
- Foundation for common software maintenance definitions and terminology
- Basis for identifying the specific SWM cost elements attributable to a given system and/or organization software maintenance effort
- Product based - system and organizational cost elements identified as those required to make changes to an operational software baseline(s)
- Cost elements represent both system allocated and non-system specific products and activities
- Flexible structure - designed to be adapted to unique system contexts and existing data structures
- The SWM-WBS is equally applicable to:
    - software maintenance estimation and planning
    - tracking software maintenance execution

Acquisition Milestone CER/SER Matrix

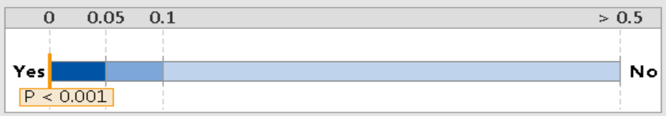
WBS Element #	MS A	MS B	MS C	Production/PPSS
1, 2 & 6 - Software Change Produce, Project Management	Analogy for cost based on system domain	Analogy for size Historical data derived proxy tables for size-based effort and cost with system domain	Development (baseline) size and build information (Program data derived formulas for effort, schedule, and cost)	MS C information plus actual maintenance data from completed releases (locally derived formulas for effort, schedule, and cost)
3 - Software Licenses (Cost of)	Analogy based on type of system and anticipated maintenance depot	Information by system type – used in analogy Post-MS B, quoted costs from vendor(s)	List of actual products with costs – license quoted costs	List of actual products with costs – license quoted costs (changes for obsolescence)
4 – Certifications & Accreditations	Analogy for cost by system domain	Analogy for cost by system domain and anticipated maintenance depot	List of actual C&As with costs (by release or annual)	List of actual C&As with costs (by release or annual)
5 – Software Maintenance Facilities	Analogy for cost by maintenance depot	Analogy for cost by maintenance depot	Budget cost (%) by depot plus extras	Actual cost (%) by depot plus extras
7 - Field Software Engineering	Analogy for cost by system domain	Analogy for cost by system domain	Analogy for cost by system domain	Analogy for cost by system domain
8 – Support Infrastructure	Analogy for cost by maintenance depot	Analogy for cost by maintenance depot	Budget cost (%) by maintenance depot plus extras	Actual cost (%) by maintenance depot plus extras

Effort CER
Engineering Super-Domain

Regression for Effort (hrs) vs ESLOC
Summary Report

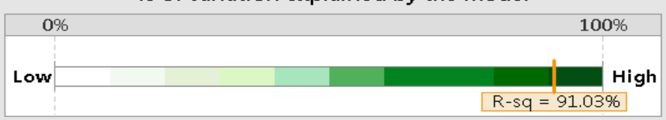
Y: Effort (hrs)
X: ESLOC

Is there a relationship between Y and X?



The relationship between Effort (hrs) and ESLOC is statistically significant ($p < 0.05$).

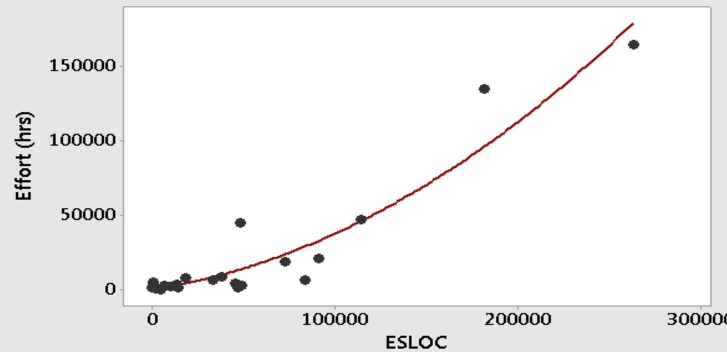
% of variation explained by the model



91.03% of the variation in Effort (hrs) can be explained by the regression model.

Fitted Line Plot for Quadratic Model

$Y = 134 + 0.1909\,X + 0.000002\,X^2$



Comments

The fitted equation for the quadratic model that describes the relationship between Y and X is:

$Y = 134 + 0.1909\,X + 0.000002\,X^2$

If the model fits the data well, this equation can be used to predict Effort (hrs) for a value of ESLOC, or find the settings for ESLOC that correspond to a desired value or range of values for Effort (hrs).

A statistically significant relationship does not imply that X causes Y.

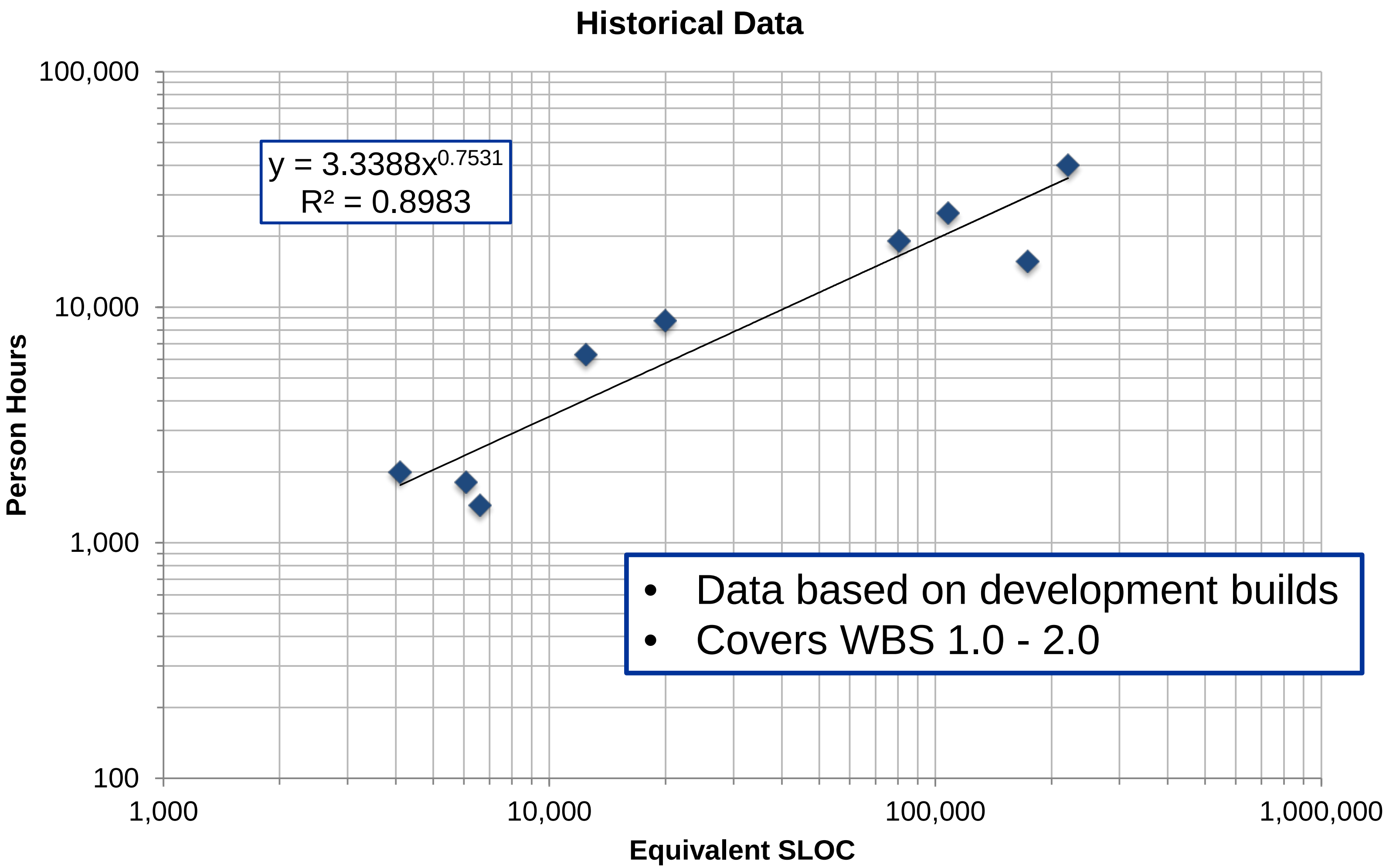
Regression analysis results for 27 Engineering (ENG) domain data points

CER - Project Data

Build	Equivalent SLOC	Hours	Start Date	End Date	Months Duration
1	173,447	15,648	01/01/08	07/10/09	18.0
1a	6,085	1,806	07/01/09	09/15/09	2.0
1b	6,609	1,441	09/10/09	11/03/09	1.0
2	108,081	25,153	06/01/09	12/15/10	18.0
2a	12,436	6,305	12/01/10	05/05/11	5.0
2b	4,106	1,994	04/01/11	10/15/11	6.0
3	220,788	40,104	08/20/11	01/09/13	16.0
3a	19,969	8,785	12/01/12	03/30/13	3.0
3b	80,575	19,105	02/01/13	09/05/13	7.0

CER Data (Equivalent SLOC, Hours) **Rhythm Data** (Start Date, End Date, Months Duration)

CER Derivation



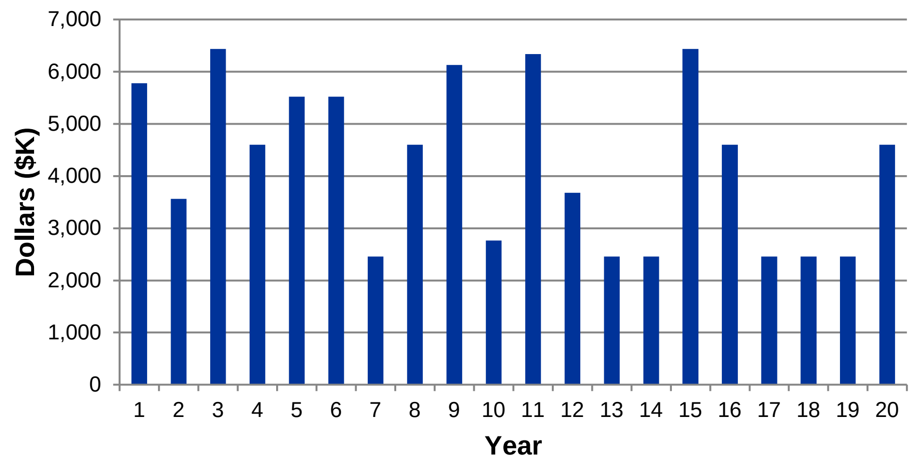
- Data based on development builds
- Covers WBS 1.0 - 2.0

Cost Projection - WBS 1.0 and 2.0

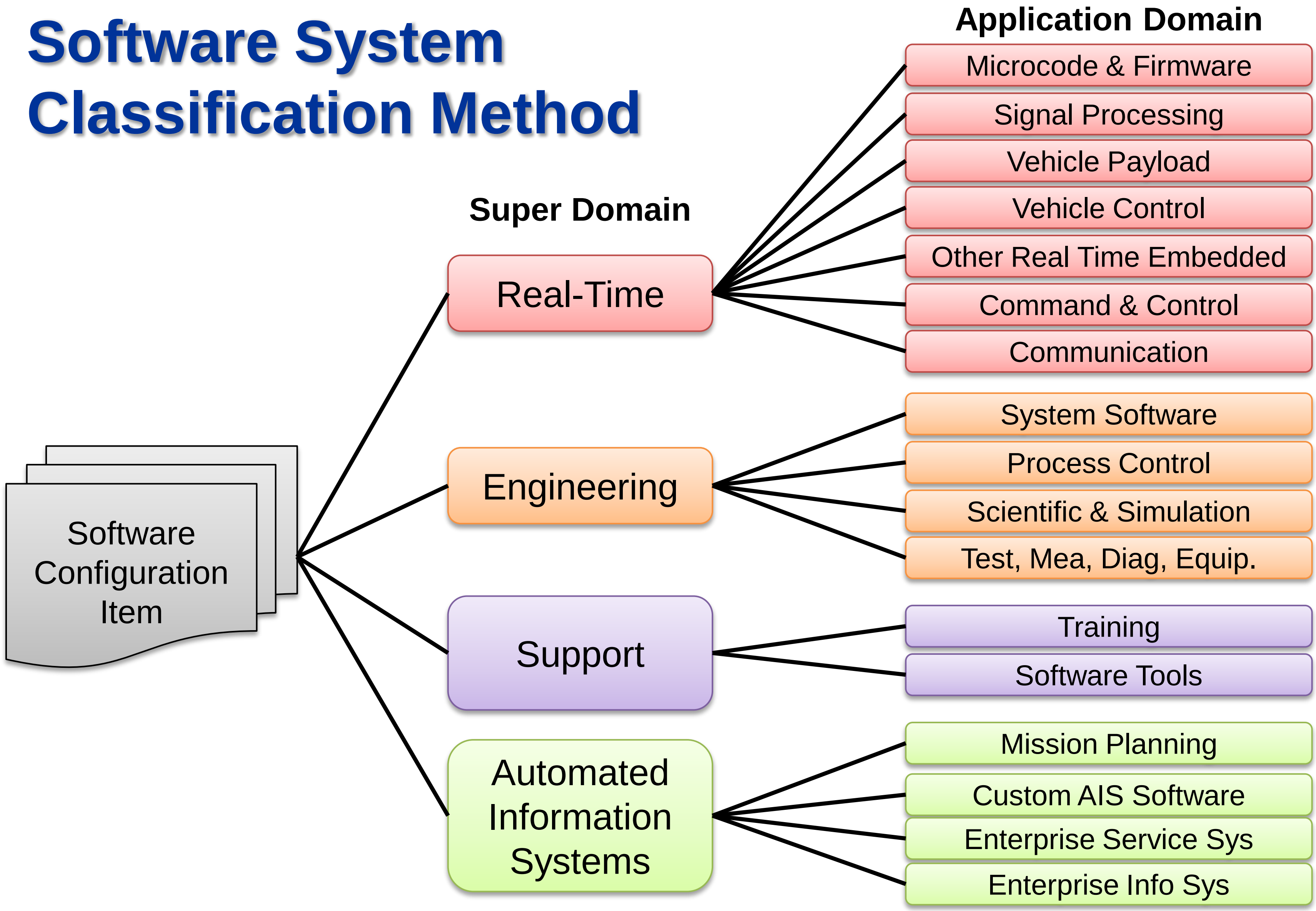
Release	SLOC	Effort (staff hours)	Cost (@ $90/hour)	Duration (months)	Monthly Burn Rate
4	175,000	87,936	$7,914,281	16	$481,707
4.1	25,000	14,678	$1,321,056	6	$204,616
4.2	25,000	14,678	$1,321,056	6	$204,616
5	200,000	99,431	$8,948,784	18	$510,857
5.1	25,000	14,678	$1,321,056	6	$204,616
5.2	25,000	14,678	$1,321,056	6	$204,616
6	200,000	99,431	$8,948,784	18	$510,857
6.1	25,000	14,678	$1,321,056	6	$204,616
6.2	25,000	14,678	$1,321,056	6	$204,616
6.3	25,000	14,678	$1,321,056	6	$204,616
6.4	25,000	14,678	$1,321,056	6	$204,616
(cont.)					

Estimate by Release

Cost Estimate by Year



Software System Classification Method



Operating Environment

- Operating Environment: In which the maintained software system operates:
  - Surface Fixed - in a system at a fixed site
  - Surface Mobile - in a system that is moved & setup
  - Surface Portable - in a handheld or portable device
  - Surface Vehicle - as part of a moving vehicle
  - Air Vehicle - as part of an aircraft
  - Sea Systems - as part of a surface or underwater boat/ship
  - Ordnance Systems - as part of a missile or rocket
  - Space Systems - as part of a spacecraft

- Manned vs. Unmanned: For the operating environment above, indicate if it is a manned or unmanned environment.

Current Efforts

Software Maintenance Information Stakeholder Requirements

- Senior Army Leadership
  - SWM requirements estimation - planning
  - SWM enterprise resource prioritization
  - System portfolio funding - execution - performance
- Multi-Project Resource and Technical Management
  - Requirements identification and prioritization
  - SWM funding and resources - planning - budgeting - execution - reporting
  - Multi-system maintenance capabilities - facilities - etc.
- System Level Project and Technical Management
  - SWM system project management
  - Change development and integration

Associated Data Requirements

- Program Level Data
  - Context information
  - Annual effort /cost data (WBS elements #3 through #8, plus total annual)
  - Organizations involved

- Release level data
  - Release context information
  - Operating environment
  - Application domain
  - Size data (those that apply)
    - Software requirements
    - External requirements
    - Source Lines of Code (SLOC)
    - Non-SLOC based size (e.g. RICE-FW, use cases, story points)
    - Software changes counts by priority (e.g. change requests, problem reports, defects)
    - IAVAs
  - Release effort / cost (WBS elements #1 and #2)
  - Schedule - start and end dates

- Details on Software Licenses
  - Right to use and maintenance

Formal Army SWM Data Collection

- Collect and evaluate SWM execution data from a wide base of Army systems - data call 22 May 2015
  - Phase 1 (3 months) - data from 5 programs from each PEO/SEC
  - Phase 2 (9 months) - data from remaining Army programs
- Evaluate the current data with respect to stakeholder information-decision requirements:
  - Availability
  - Integrity
  - Usability
- Update the existing SWM estimation methodology and refine the underlying CERs

Software Maintenance Data Evaluation

SWM Data Evaluation

Availability

- Completeness of required data set
- Underlying SWM business and technical processes are well enough defined to produce objective data on a periodic and/or event driven basis
- IT systems and tools exist to enable systematic and timely data collection

Integrity

- Data are derivatives of actual SWM technical and management processes
- All data (measures) are explicitly defined - measurement contexts are known
- Cost data is directly correlated with the WBS defined output products and activities
- Data is consistent - methods exist to address system conflicts (normalization)

Usability

- Data is aligned with stakeholder decision information needs
- Data can be objectively characterized and interpreted
- Mapping and aggregation structures and methods exist to combine data
- Potential emerging information requirements have been considered

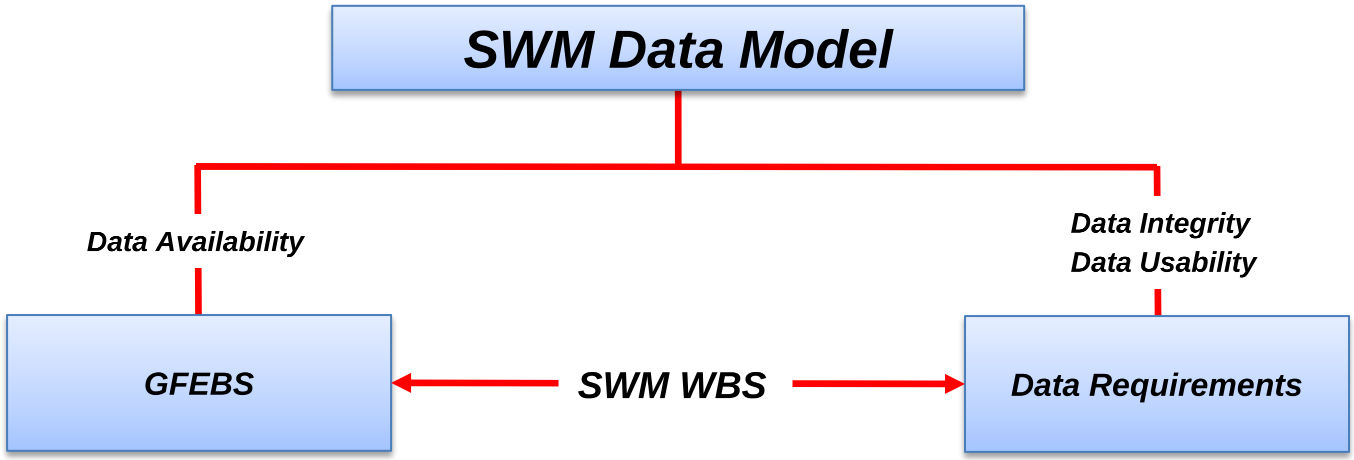
SWM Data Evaluation - Program Level

		Initial Program Overall			Detailed Program Assessment						
Lead Gov't Org	Program	WBS 3-8	Definable Maint. Process	Total Program Effort/Cost	License Management (WBS-3)	C&A Support (WBS-4)	System Facilities Management (WBS-5)	Sustaining Engineering (WBS-6)	Field S/W Engineering (WBS-7)	Operational Management (WBS-8)	License Costs
Org 1	Pgm 1	R	R	R	N/A	R	R	R	R	R	N/A
Org 2	Pgm 2	Y	G	G	N/A	N/A	G	G	G	R	N/A
Org 2	Pgm 3										
Org 2	Pgm 4										
Org 2	Pgm 5										
Org 3	Pgm 6										
Org 3	Pgm 7	R	G	G	N/A	Y	R	R	Y	R	N/A
Org 3	Pgm 8	R	G	O	R	R	R	R	R	R	G
Org 3	Pgm 9	O	G	B	N/A	O	O	O	O	R	N/A
Org 3	Pgm 10	Y	R	O	R	G	G	R	R	G	G

SWM Data Evaluation - Release Level

				Initial Release Overall		Detailed Release Assessment						
PID	Lead Gov't Org	Program	Release	CDR Usability	SER Usability	Size: Requirements	Size: Number (#) of External Interfaces	Size: SLOC	Size: non-SLOC	Size: Changes	Effort (WBS-1 &2)	Schedule (WBS-1&2)
	Org 1	Pgm 1	Build 3.0 C4a	R	R	G	G	R	N/A	N/A	R	R
	Org 1	Pgm 1	Build 3.0 C4	R	R	G	G	Y	N/A	N/A	R	R
	Org 2	Pgm 2	v3.10.4 (PoR)	G	R	G	N/A	G	N/A	G	G	N/A
	Org 2	Pgm 2	v3.9.200 (DOE)	G	G	G	N/A	G	N/A	G	G	G
	Org 2	Pgm 2	v3.10.210 (SOCOM)	G	G	G	N/A	G	N/A	G	G	G
	Org 2	Pgm 2	v3.10.3 VIS/IR (SOCOM)	G	G	G	N/A	G	N/A	G	G	G
	Org 2	Pgm 2	v3.9.3 (Abrams)	G	G	G	N/A	G	N/A	G	G	G
	Org 2	Pgm 3										
	Org 2	Pgm 4										
	Org 2	Pgm 5										
	Org 3	Pgm 6										
	Org 3	Pgm 7	999	R	R	R	R	R	R	R	R	R
	Org 3	Pgm 8	999	R	R	R	R	R	R	R	R	R
	Org 3	Pgm 9	ED 8.0	R	O	R	R	R	R	O	O	G
	Org 3	Pgm 10	5.3.1-2			G	G	G	R	G	B	B
	Org 3	Pgm 10	5.4.0-4			G	G	G	R	G	B	B

Systems - Data Integration



Automated Data Collection
- Cost Data ($ - Effort)
- Technical Data (Size - Priority - etc.)
- Product Schedule Data

Aggregation Structures/Mappings
Cost Object Alignment
Plans - Actuals
Flexible User-defined Data Fields
Consistent Implementation
Governance

Process Derived Data
- Cost Data ($ - Effort)
- Product Schedule Data
- Technical Data
- System Context Data

Cost-Product Correlations
Aggregation Structures/Mappings
Organizational Cost Allocations to System
Data Definitions
Data Collection Procedures
Analysis and Evaluation
Stakeholder Information Needs

Refinement of CERs

- Refine the underlying SWM CERs - calibrate to application domains and operating environments
- Update the estimation models/methodology with respect to structure and application
- Define the requirements/plan for implementing an Army SWM information infrastructure - focus on multi-level decision information needs
- Adapt GFEBS to support systemic SWM data collection

What We Have Learned

- Estimating software maintenance is much more difficult than estimating software development:
  - Complex cost, funding, and management constructs
  - Lack of a consistent data environment - execution data not used to manage
  - Focus on system/organizational funding - not cost of output products/services
  - Lack of visibility into leveraged contractor efforts and expenditures
- Two significant cost categories:
  - Fixed infrastructure - maintenance of capability costs
  - Variable change driven software modification costs
- What is paid for and what is done are two different things - SWM task volatility
- The emerging estimation methodology more closely aligns with the SWM work that is actually being accomplished
- If requirements continue to grow beyond the projected SWM budgets - we will need to be much better at estimating, allocating, and tracking SWM costs

Contact Information

James Judy
U.S. Army Office of the Deputy Assistant Secretary of the Army for Cost & Economics (ODASA-CE)
5701 21st Street
Fort Belvoir, VA 22060-5546
(703) 697-1612
james.m.judy.civ@mail.mil

John McGarry
U.S. Army RDECOM - ARDEC
Quality Engineering & System Assurance
RDAR-QES-A
Picatinny Arsenal, NJ 07806-5000
(973) 724-7007
john.j.mcgarry4.civ@mail.mil

James P. Doswell
U.S. Army Office of the Deputy Assistant Secretary of the Army for Cost & Economics (ODASA-CE)
5701 21st Street
Fort Belvoir, VA 22060-5546
(703) 697-1572
James.p.doswell.civ@mail.mil

Cheryl Jones
U.S. Army RDECOM - ARDEC
Quality Engineering & System Assurance
RDAR-QES-A
Picatinny Arsenal, NJ 07806-5000
(973) 724-2644
cheryl.l.jones128.civ@mail.mil

Additional Information

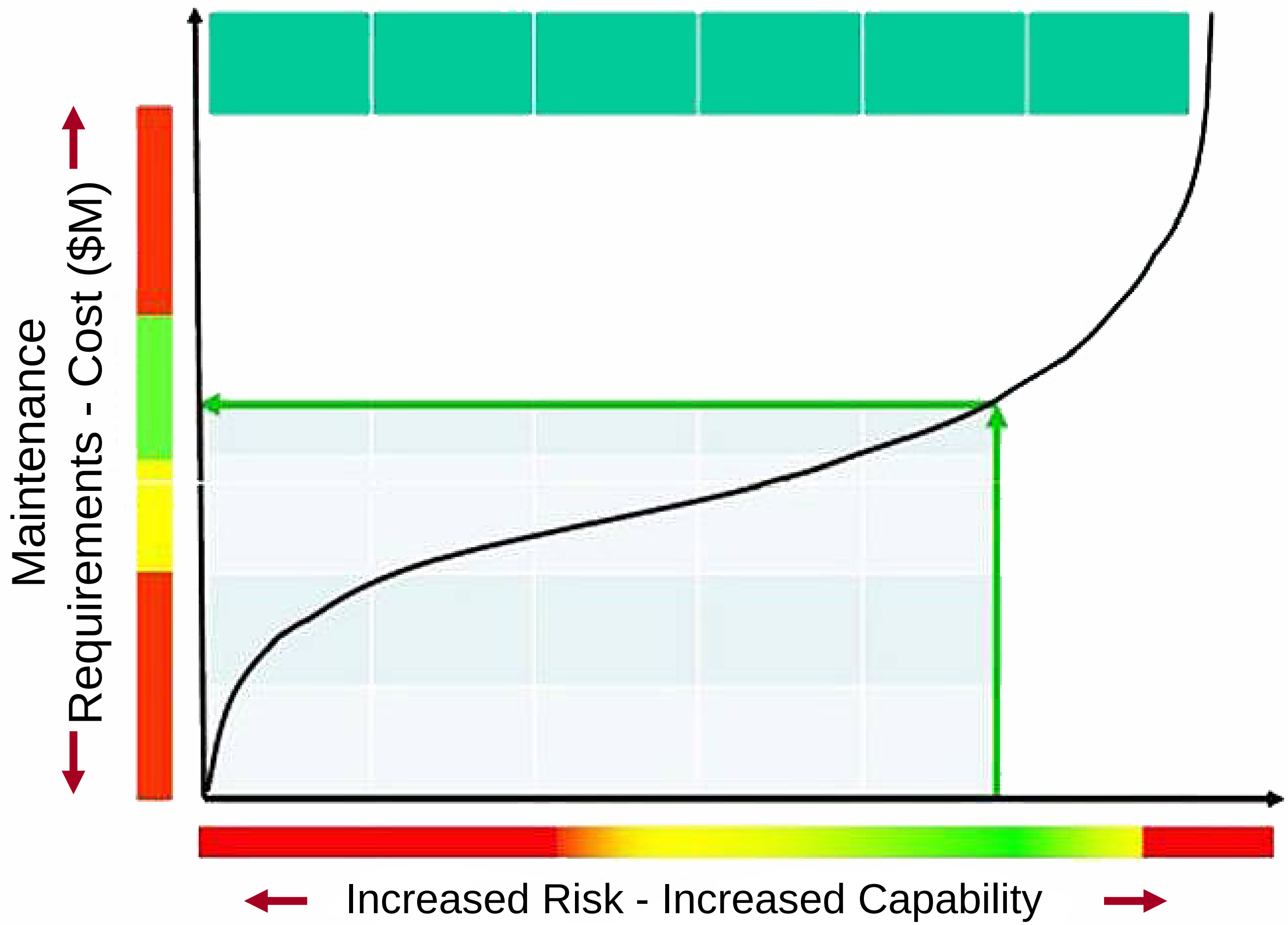
Army SWM Stakeholders

- SWM Enterprise Managers
  - ASA(ALT)
  - AMC
  - Army G4/G48/G8
  - DASA-CE
  - Army Cost Management Executives
  - Army Software IPT
- System Acquisition PEOs/PMs
  - PEO Ammo
  - PEO Aviation
  - PEO C3T
  - PEO CS&CSS
  - PEO EIS
  - PEO GCS
  - PEO IEW&S
  - PEO MS
  - PEO SOLDIER
  - PEO STRI
  - PEO Chem/Bio
  - JM&L LCMC
  - MDA
- SWM Technical Organizations
  - SEC AMRDEC
  - SEC CECOM
    - Ft. Sill
    - Ft. Huachuca
  - SEC TARDEC
    - Picatinny Arsenal
- SWM System Technical Teams
  - SWM System Leads
  - Facility Leads
  - Focused Support Assets
- SWM Contractors
  - SWM Project Managers
  - SWM System Technical Leads
- Others
  - OSD/CADE
  - Government Contracting Organizations
  - Congress
  - GAO/IG

Army Software Maintenance Data Environment

- Minimal enterprise level SWM governance/policy (DOD, Army, etc.) - Diverse business and technical processes defined by each stakeholder organization
- Diverse/limited technical and management capability of stakeholder organizations
- Inconsistent SWM resource allocation processes - requirements models - Emphasis on justifying funding requirements - not on performance based execution
- LOE management structures - LOE resourcing strategies
- Multiple funding sources/types applied to the same system/SWM products
- Discordant SWM maintenance cost accounting accrual structures (system, functional, organizational, etc.)
- Volatile SWM change requirements during execution
- Lack of reported contractor performance data (cost/schedule/product output)
- Non-aligned cost, resource, and software technical SWM - Many “stovepiped” data parameters
- Funding/Cost not tied to SWM product outputs

Objective Cost-Capability Trades



"It's All About the Money", Dr. Chien Huo, CAPE, November 2011

SWM Work Breakdown Structure

1.0 Software Change Product - Effort and cost to implement software changes including all effort and cost associated with defining, allocating, generating, integrating, and testing software changes for an operational software product or system, for a release. Includes addressing/correcting errors and IAVAs, including requirements analysis, design, implementation, and software testing.

2.0 System Project & Technical Management - Effort and cost associated with system specific software maintenance project and technical management for this release. These activities include planning, execution management, configuration management, resource management, contracting and measurement reporting.

3.0 Software Licenses - Effort and cost associated with the procurement and renewal of software licenses for NDI, COTS, GOTS, or open-source software, for the maintenance facility and the deployed systems.

4.0 Certification and Accreditation - Effort and cost associated with verifying a software system against externally defined domain performance criteria for this release. These activities include managing certification and accreditations, such as Security, RMF, DIACAP, IAVA, Safety, Airworthiness, and Networthiness.

5.0 System Facilities - Effort and cost associated with establishing, operating and sustaining software development assets / workstations, integration / test facilities, labs, and support equipment and tools.

6.0 Sustaining Engineering - Effort and cost associated with engineering (e.g. system specific test support, investigations, PM support) and user support (e.g. user training, help desk, software delivery).

7.0 Field Software Engineering - Effort and cost associated with the on-site support of a deployed software product or system in its operational environment. FSE duties may include on-site technical assistance, problem troubleshooting, software installation, operational assistance, or on-site training.

8.0 Operational Management - Effort and cost allocated to pay for non-system specific OM resources that are allocated to a system. OM resources are those associated with establishing and operating the organizational infrastructure required to implement common software maintenance business and technical processes across multiple software systems, including operations, organization mgmt., personnel mgmt., financial mgmt., information mgmt., process mgmt., or change mgmt..

Software Maintenance Change Product Definition

- SWM WBS 4.4c allows for multiple definitions of the software maintenance products, specifically the software release or the individual software change
- How the software product is defined is driven by:
  - The available data (a function of the instantiated program processes)
  - The specific information/analysis needs (planning, release development and test, execution tracking, etc.)
- These product definitions do not conflict - SWM releases and groupings of changes into “capability sets” and other composite products are all aggregations of one or more individual software changes
- The software release is the primary SWM output product for Army systems

Software Maintenance Change Product

- WBS 4.4c Element 1.0 includes three primary sub-elements:
  - 1.1 Change Requirements - identify, prioritize, and group individual changes and allocate the changes to capability sets and software releases
  - 1.2 Change Development - design, code, unit test, and integrate individual changes
  - 1.3 Baseline Integration and Test - change integration, interface testing, system software testing, and QA/IV&V
- These cost elements apply equally to an individual software change or a software release comprised of multiple changes

Information Requirements Overview

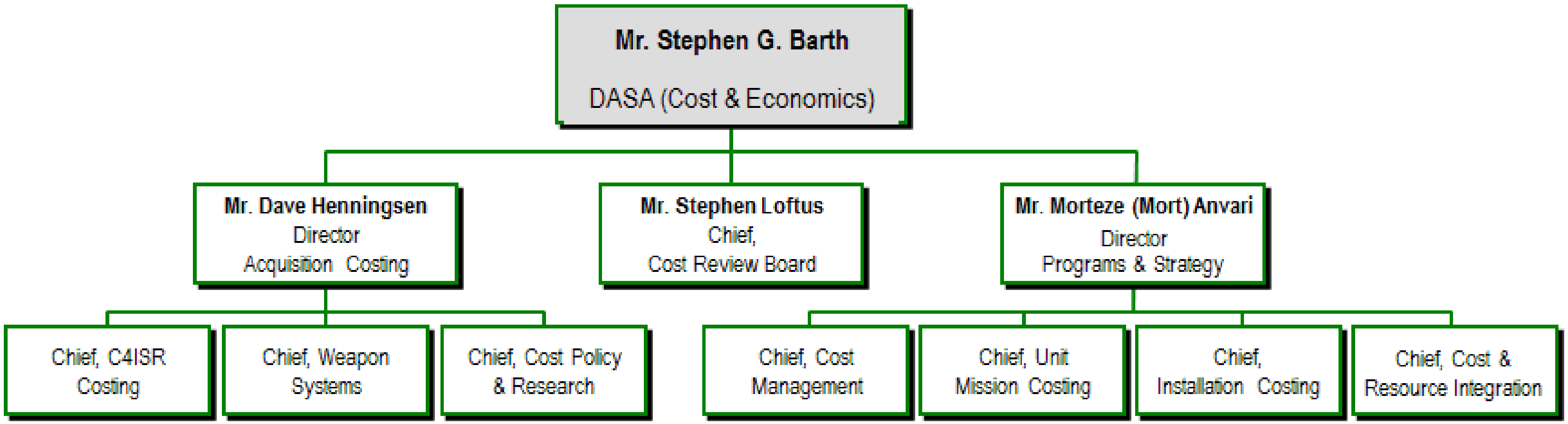
- Software maintenance information requirements for all Army stakeholders are directly tied to enterprise, organization, and system SWM objectives.
- There are multiple SWM stakeholder data/information perspectives - both common and unique information requirements can be identified across the SWM stakeholder base.
- The defined information requirements drive the SWM data that needs to be collected and analyzed.
- The availability, integrity, and usability of the data is a function of the capability of the implemented SWM technical and business processes.
- In general, SWM data is generated at the system and organizational levels and is aggregated upwards

Challenges with Sizing and Estimating Enterprise Information Systems

Mr. Dave Fersch
Office of the Deputy Assistant Secretary of the Army - Cost & Economics (ODASA-CE)
and
Dr. Chris Miller
Quantitative Software Management, Inc. (QSM)

Office of the Deputy Assistant Secretary of the Army - Cost & Economics (DASA-CE)

Cost and Economics Organizational Alignment



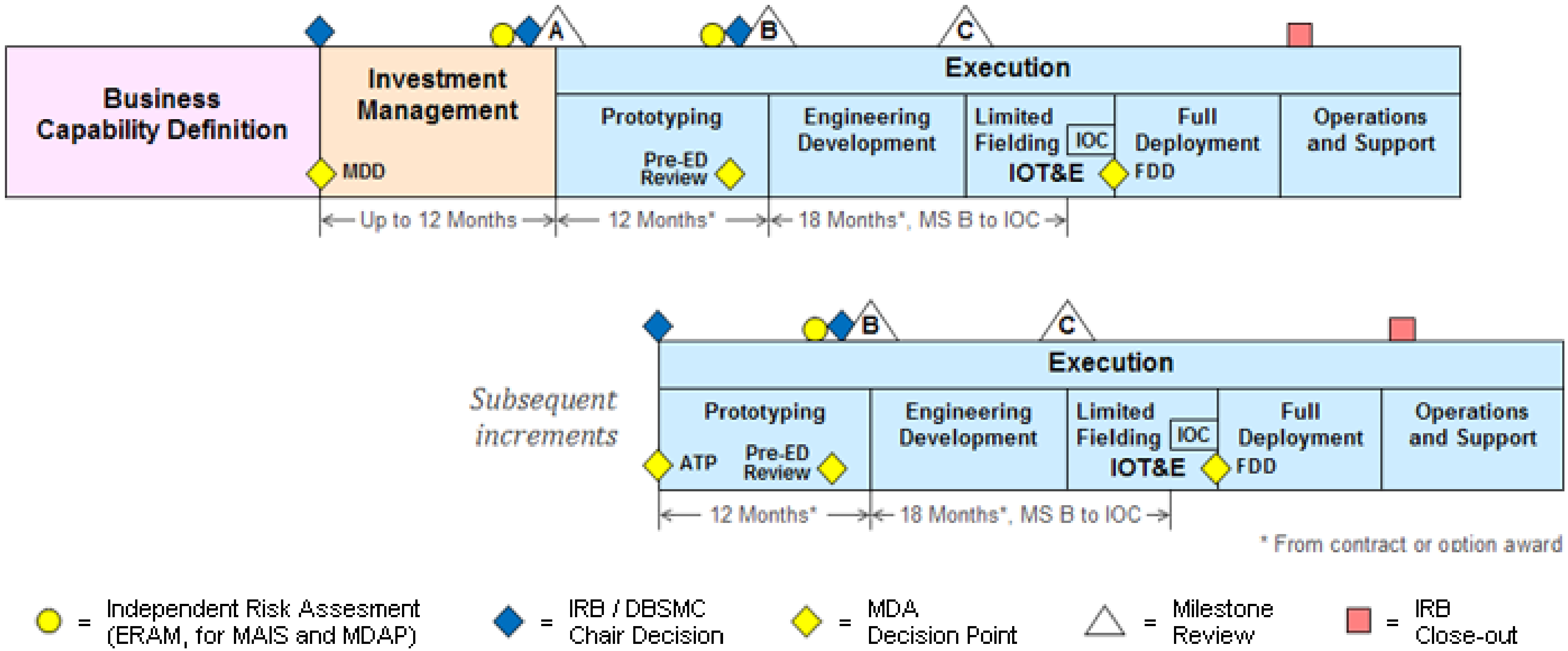
Mission

Provide the Army decision-makers with cost, performance and economic analysis in the form of expertise, models, data, estimates and analyses at all levels.

Vision

Innovative and impartial center of excellence relied upon by Army Leadership. Dedicated to cost forecasting, analyses and performance management through trained people.

Business System Acquisition Framework

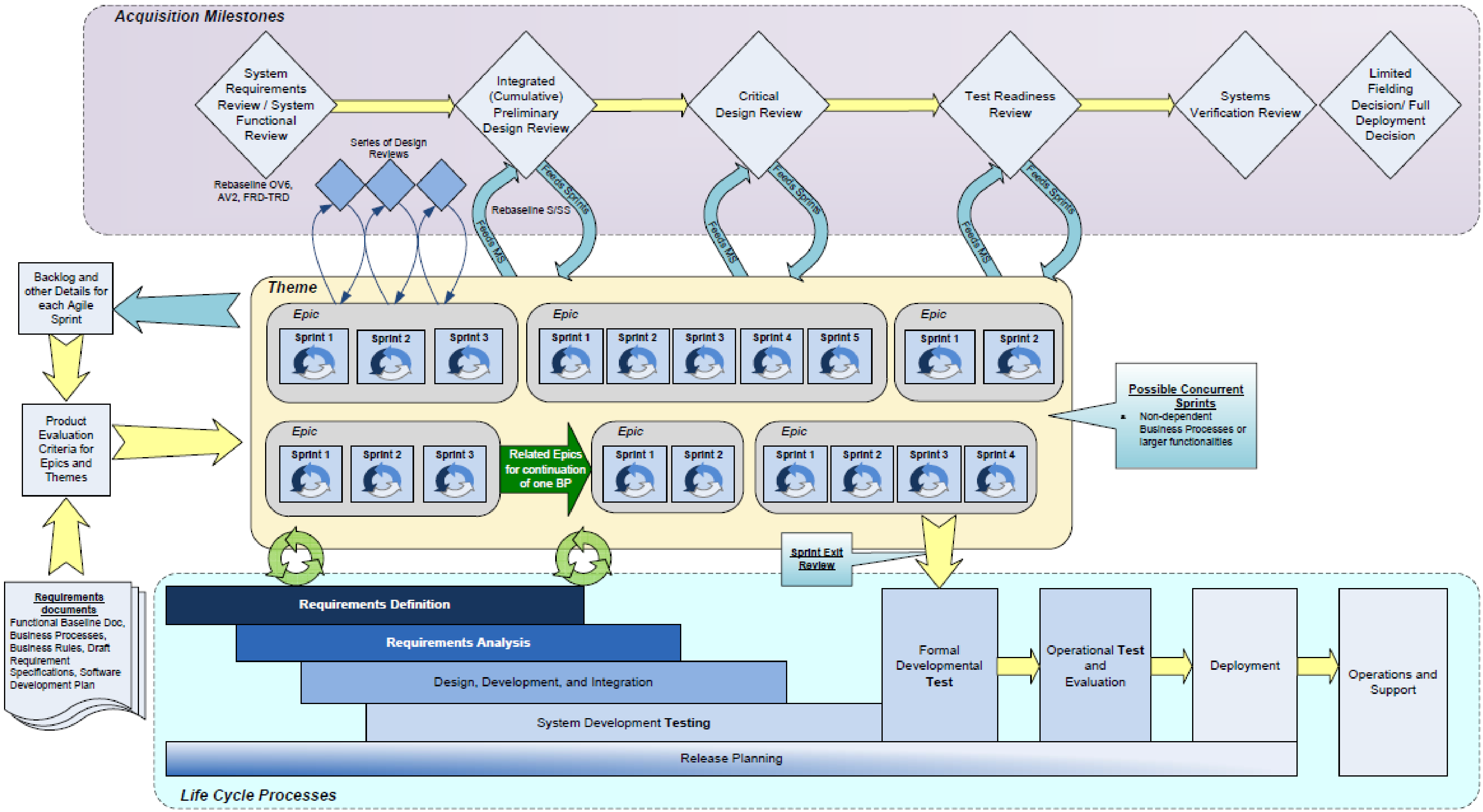


The Business Capability Lifecycle model depicts the process for the acquisition of Defense Business Systems (DBS) and the framework for the definition, development, testing, production, deployment, and support of DBS. The principles of the model apply to the initial release and subsequent increments.

— DTM 11-009, *Acquisition Policy for Defense Business Systems*

QSM® Quantitative Software Management

What we are seeing…



Testing of Key Estimation Concepts

Estimation Key Concepts

- Objective: use a measure that allows correlation of size and effort (i.e., a good estimator for effort)
- Objective: Select a size measure that may be used to estimate across the life cycle
- Objective: use practical sizing methods based on the software development process & artifacts

EIS Challenges

- Significant effort is expended on unplanned course corrections versus planned development activities
- EIS software evolves during development & operations; typically the end system is not the system initially envisioned or funded
- COTS integration involving package configurations and extensions do not use conventional size measures
- Agile development limits early lifecycle sizing beyond high level requirements (hindering detailed sizing prior to post-MS B)

EIS Characteristics & Size Usage

	Proj Alpha	Proj FED	Proj LOG	Proj MIL	Proj Golf
Software Development Characteristics:					
• Lifecycle Phase	MS A	FD	MS C	MS B	FDD
• Waterfall					
• Incremental		X	X	X	x
• Agile	?			x	X
• Core COTS Product	Undecided	SAP	SAP	PeopleSoft	SAP
Size Measure Usage (as provided to DASA-CE):					
• ESLOC					
• Function Points					
• RICEFW / Configurations	**X**	**X**	**X**	**X**	**X**
• Requirements	x				

QSM®
Quantitative Software Management

Package Implementation - Business Processes Configurations & RICEFW Objects

ODASA Cost & Economics

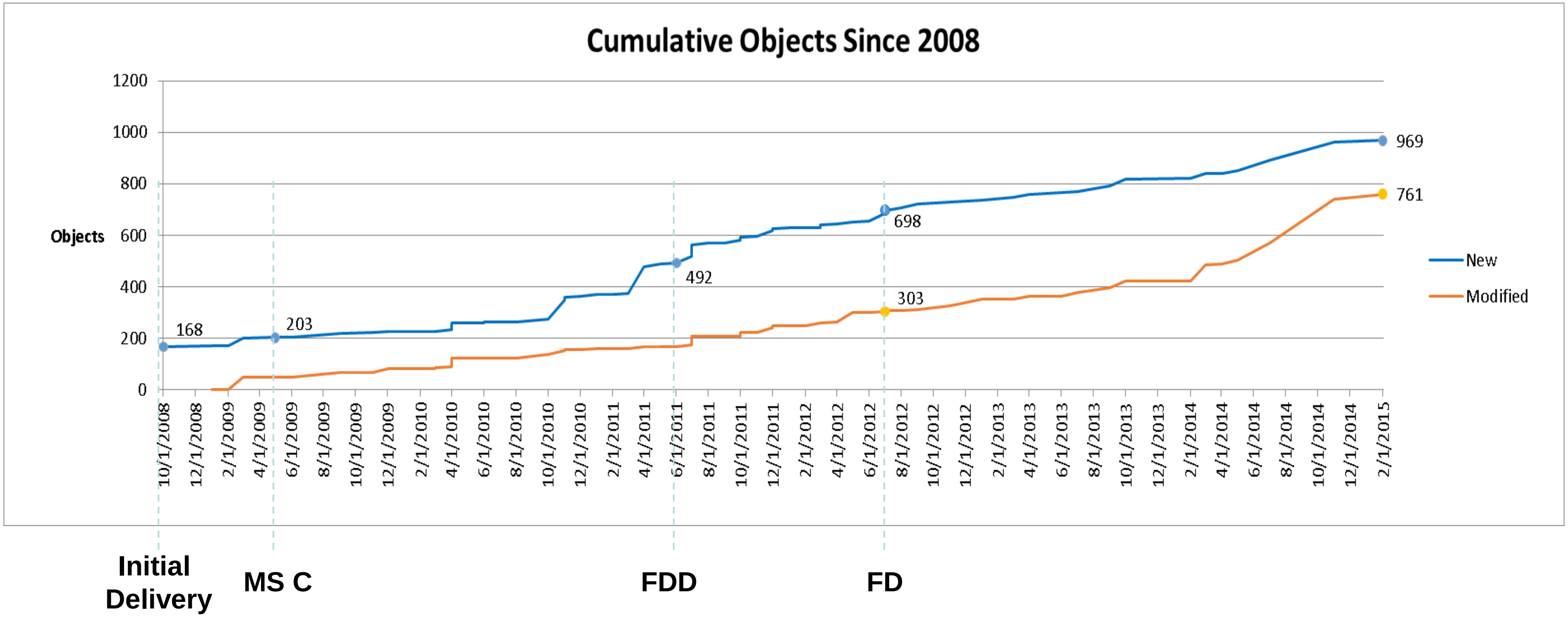
- Count the number (size) of business processes delivered by the package (i.e., configurations)
  - High Level Business Processes or "Scenarios"
  - Detailed Business Processes
- Identify and count the custom development portion needed
  - RICEFW Objects
    - Reports
    - Interfaces
    - Conversions
    - Extensions
    - Forms
    - Workflows

	Gear	R1.1	IU	R1.2	IU
BUSINESS PROCESSES					
Simple Business Scenarios	300		0		0
Average Business Scenarios	600		0		0
Complex Business Scenarios	1200	3	3600	3	3600
Total		3	3600	3	3600
CONFIGURATION DESIGNS					
Simple Configuration Designs	80	152	12160	17	1360
Average Configuration Designs-160	160	195	31200	36	5760
Complex Configuration Designs	320	29	9280	4	1280
Total		376	52640	57	8400
RICE					
Simple Reports	100		0		0
Average Reports	200		0		0
Complex Reports	300		0		0
Simple Interfaces	320	11	3520		0
Average Interfaces	620	14	8680	5	3100
Complex Interfaces	1520	11	16720	19	28880
Simple Conversion	100		0		0
Average Conversions	200		0		0
Complex Conversions	300		0		0
Simple Extensions	100	1	100		0
Average Extensions	500		0		0
Complex Extensions	1000	8	8000	10	10000
Simple Forms	100		0		0
Average Forms	200		0		0
Complex Forms	300		0		0
Simple Workflow	100		0		0
Average Workflow	500	1	500		0
Complex Workflow	1000		0		0

Initial Sizing and Size Stability

- Limited availability of quantifiable system artifacts beyond high-level requirements and core business processes at Milestones A & pre-B
- Significant number of course corrections:
  - Policy changes/mandates
  - Evolving external system interfaces
  - User-driven changes (extending functionality, improving performance and defect resolution)
  - Cybersecurity
- Evolving end product beyond initial deployment
  - For example: new DoD Directive states an interfacing system will be retired and now the functionality needs to be provided by System X

EIS Project Size Growth Example



- Initial cost estimate assumed no 'new' RICEFW object growth and 25% break/fix for modifications post Full Deployment
- RICEFW object growth continues (271 since FD) and actual break/fix to date is 65%

Challenges during Program Cost Estimate Reconciliation Meetings

- Lack of historical data (analogous data points)
- Traditional sizing measures don’t translate; willingness to use RICEFW objects
- Lack of understanding and experience with RICEFW objects

Delta between DASA-CE independent estimate and Program estimate often comes down to:

- Assumptions (engineering approach, reuse, funding sources)
- Size (based on engineering artifacts; normalization)
- Historical data (basis for CERs)
- Estimation approach

Critique of Size Measures for EIS

	Strengths	Weaknesses	Opportunities for improvement
Business Requirements	Available early in the life cycle	Highly variable to effort	Use of non-DoD historical data (i.e., analogous)
Business Processes	Core Functionality based	Highly variable to effort	Historical data; metadata
ESLOC	Code counting tools and robust definitions; minimal counting variation	Not natural by-product of EIS software development environment	Establish definitions for size normalization of EIS work products
Function Points	Counting standards & definitions, minimal variation	Lack system definition at early milestones; training investment	Invest in function point counting of analogous and target system
RICEFW / Configurations	Natural by-product of software end product	Not well defined (lack of standardized counting guidance)	Counting guidance and standardization
Agile User Stories / Story points	Natural by-product of the software development process & end product	Definition of a User story varies; Use of Story Points is scarce	Increase usage of analogous historical data

Sizing Observations

- EIS/ERP/Agile implementations are introducing terminology (e.g., themes, workstreams) affecting standardized data collection and hindering future cost estimation effectiveness
- RICEFW Objects lack of definition and counting guidance/standards cause inconsistent counting results
- Requirements: Most COTS ERP providers don't have their documentation written to DoD Standards (i.e., lack of a robust system requirements specifications prohibits counting 'shalls' in a manner meaningful for cost estimation sizing purposes)

The cone of uncertainty applies to EIS

Final thoughts…

- Large variation in definition and quantifying size measures leads to ineffective cost estimating relationships at Milestones A and B (prior to systems integrator contract award)
- Too often sizing (and cost estimation) appears as an afterthought to other acquisition activities
- Reported size measures change during development and deployment
  - Requirements → RICEFW → Use cases → Releases
- Witnessing significant EIS system size growth post Full Deployment (FD) driving added costs in either unplanned acquisition or unplanned maintenance

"See, this is why, I like modern architecture. The houses are too new to have ghosts" - *Gabriella Pierce*

Questions?

Mr Dave Fersch
ODASA-CE
Building 216, Fort Belvoir, VA
david.r.fersch.civ@mail.mil

Dr. Christopher L Miller
QSM, Inc.
2000 Corporate Ridge, Suite 700
McLean VA 22102
christopher.miller@qsm.com

Your Preferred Source for
IT Acquisition Across the DoD

DoD Enterprise Software Initiative

Presentation to the Software and IT Cost IPT

August 12, 2015

Floyd Groce
IT Strategic Sourcing Lead
DoD ESI Co-Chair
Enterprise Services Team
Department of the Navy CIO

www.esi.mil

Agenda

- *DoD ESI Background*
- *What's Coming – Statute, Policy, and Process Changes*
- *DoD ESI Commercial Software Licensing Training*
- *Questions*

DoD ESI Background

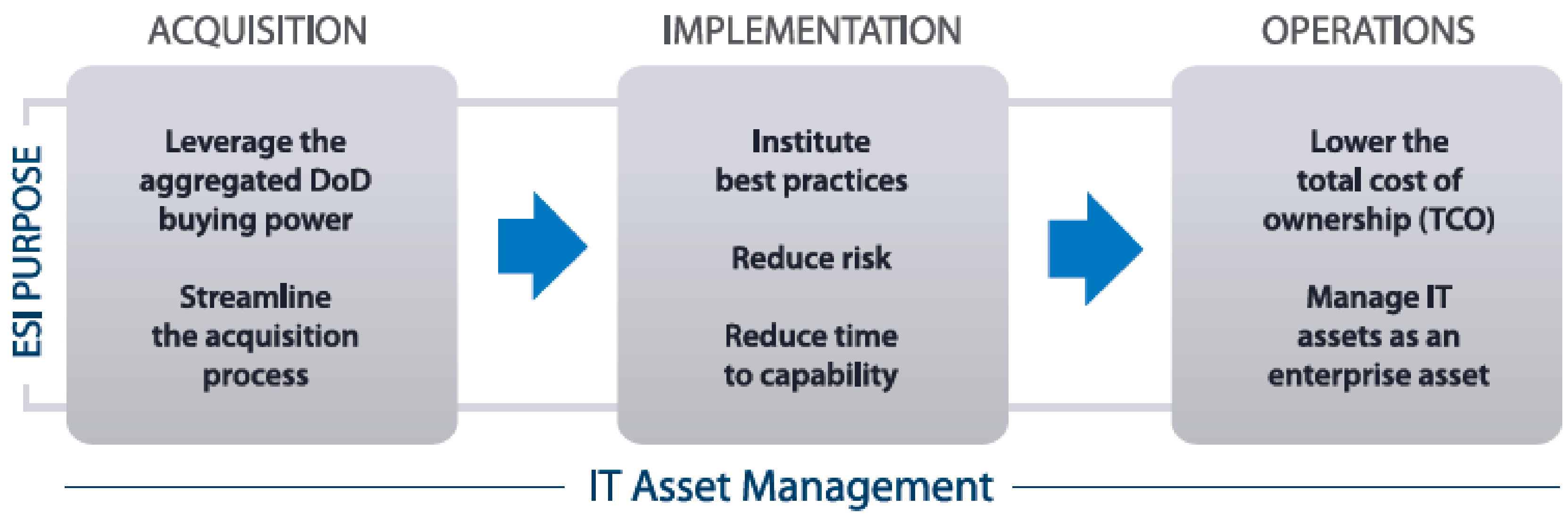
Federal Software Spend: At a Glance

- *80% of the obligations flow through GSA Schedule 70 (61%) and NASA SEWP (19%)*
- *Another 18% flows through 7 other contracting solutions*
- *There are over 44,000 transactions with an average spend per transaction of $126K*
- *10 Agencies comprise 73% of the Software obligations*
- *Agencies buy and manage their licenses in a decentralized way, so inventory management is inconsistent*
- *Agencies often overbuy (e.g., accept bundles of software when only a fraction of the functionality is needed)*
- *Terms and conditions vary widely - even across similar vehicles*
- *Pricing and other critical information is not shared*

Source: FY14 FPDS-NG

What is DoD ESI?

- ***Joint initiative to save time and money on acquisition of commercial software, IT hardware, and services***
- ***Executive Sponsor: DoD CIO***
- ***Goals***
  - ***Save time, effort, and money***
  - ***Target DoD Customer Needs and Efficiencies***
  - ***IT Asset Management***



What is DoD ESI?

- ***Team Composition:***
  - *Army, DON, Air Force, DLA, DISA, OSD*
- ***Operations:***
  - *Award enterprise agreements for IT products and services*
  - *Implement unified vendor, strategic sourcing and contract management strategy with leading IT vendors*
  - *Use an agile, low overhead model executed through Software Product Managers (SPMs) in four DoD Components*
  - *Work closely with the OMB and GSA Software Center of Excellence (formerly SmartBUY) to optimize IT acquisition policy and implement IT Strategic Sourcing and IT Category Management within DoD*
- ***Results:***
  - *Over 70 DoD ESI agreements representing with over 30 OEM publishers*
  - *Over $6 billion cost avoidance since inception*
  - *Improved IT asset visibility of DoD ESI suppliers through Reports of Purchases through all DoD ESI agreements*
  - *More efficient acquisition processes for ESA users*

DoD Regulations/Policies

DoD Instruction 5000.02 dated 7 Jan 2015

"When acquiring commercial IT, Program Managers must consider the DoD ESI, Federal Strategic Sourcing Initiative procurement vehicles, and Defense Component level Enterprise software Licenses."

DFARS Subpart 208.74

"Departments and agencies shall fulfill requirements for commercial software and related services, such as software maintenance, in accordance with the DoD Enterprise Software Initiative (ESI)."

DoD ESI - Providing Value to the Enterprise

Terms & Conditions
- Software Licenses
- Maintenance & Support
- Integration Services
- Cloud/SaaS

Market Research
- Industry Knowledge
- Trends in Technology
- Product Uniformity
- License Models

Training
- Classroom
- eLearning
- Webinars
- Workshops

Stakeholder Outreach
- Strategic Communication
- Awareness
- Education
- Sponsorship

Tools
- White Papers
- Checklists
- Web Toolkits
- Templates

Lessons Learned
- Best Practices
- Knowledge Sharing
- IT Asset Visibility
- SPM Support

Pricing
- Benchmarking
- Best Value Toolkit
- TCO Tools
- Portal

What’s Coming – Statute, Policy, and Process Changes

Legislative - Expect Continuation of Legislative Interest

- ***NDAA 2012***
  - ***Section 2867***
    - ***Waivers for IT Investments tied to Data Centers***
- ***NDAA 2013***
  - ***Section 937, "Software Licenses of the Department of Defense"***
    - ***Mandates DoD CIO, in consultation with Military Department and Agency CIOs, plan and develop inventory of selected software***
- ***NDAA 2014***
  - ***Section 935 – "Additional Requirements Relating to the Software Licenses of the Department of Defense"***
    - ***Mandates inventory of every software title on which a Military Department spends $5 million or more in a year***

Legislative - Expect Continuation of Legislative Interest

- ***NDAA 2015 –***
- ***Creates new Under Secretary of Defense for Business Management and Information***
- ***Federal Information Technology Acquisition Reform***
  - ***Section 831. Chief Information Officer Authority Enhancements***
  - ***Section 832. Enhanced Transparency and Improved Risk Management in Information Technology Investments***
  - ***Section 833. Portfolio Review***
  - ***Section 834. Federal Data Center Consolidation Initiative***
  - ***Section 835. Expansion of Training and Use of Information Technology Cadres***
  - ***Section 836. Maximizing the Benefit of the Federal Strategic Sourcing Initiative***
  - ***Section 837. Government-wide Software Purchasing Program***

Office of Management and Budget (OMB)

- *OMB Memo M-15-14 of June 10, 2015, Management and Oversight of Federal Information Technology*
  - *Section E: Information Technology Acquisition Initiatives*
    - *IT Acquisition Cadres*
    - *Category Management and the Federal Strategic Sourcing Initiative (FSSI)*
    - *Government-wide Software Purchasing Program*
- *OFPP Memo of 2 Dec 2014, "Transforming the Marketplace: Simplifying Federal Procurement to Improve Performance, Drive Innovation, and Increase Savings*
- *Federal Strategic Sourcing:*
  - *Adopt/Implement Category Management*
  - *Enterprise Software Category Team (ESCT)*
  - *IT Category Manager*
  - *Common Acquisition Platform (CAP)/Acquisition Gateway*
    - *Software Category Hallway*
    - *Prices Paid*

Sales Report Key Field Summary – Input to Database

Sales Report Field	Use
Ordering Activity Delivery Order #	Contract/Credit Card ID
End User Service	Agency
End User Ship-To Activity (Government)	Address of Ordering Activity
Product Description	Description entered by Reseller
Publisher P/N	Publisher's Part Number
ESA Unit Price	Negotiated Price on BPA
Unit Price on Order	What customer actually paid
Ordering Activity	Contracting Office

DoD ESI Commercial Software Licensing Training

DoD ESI Training Approach

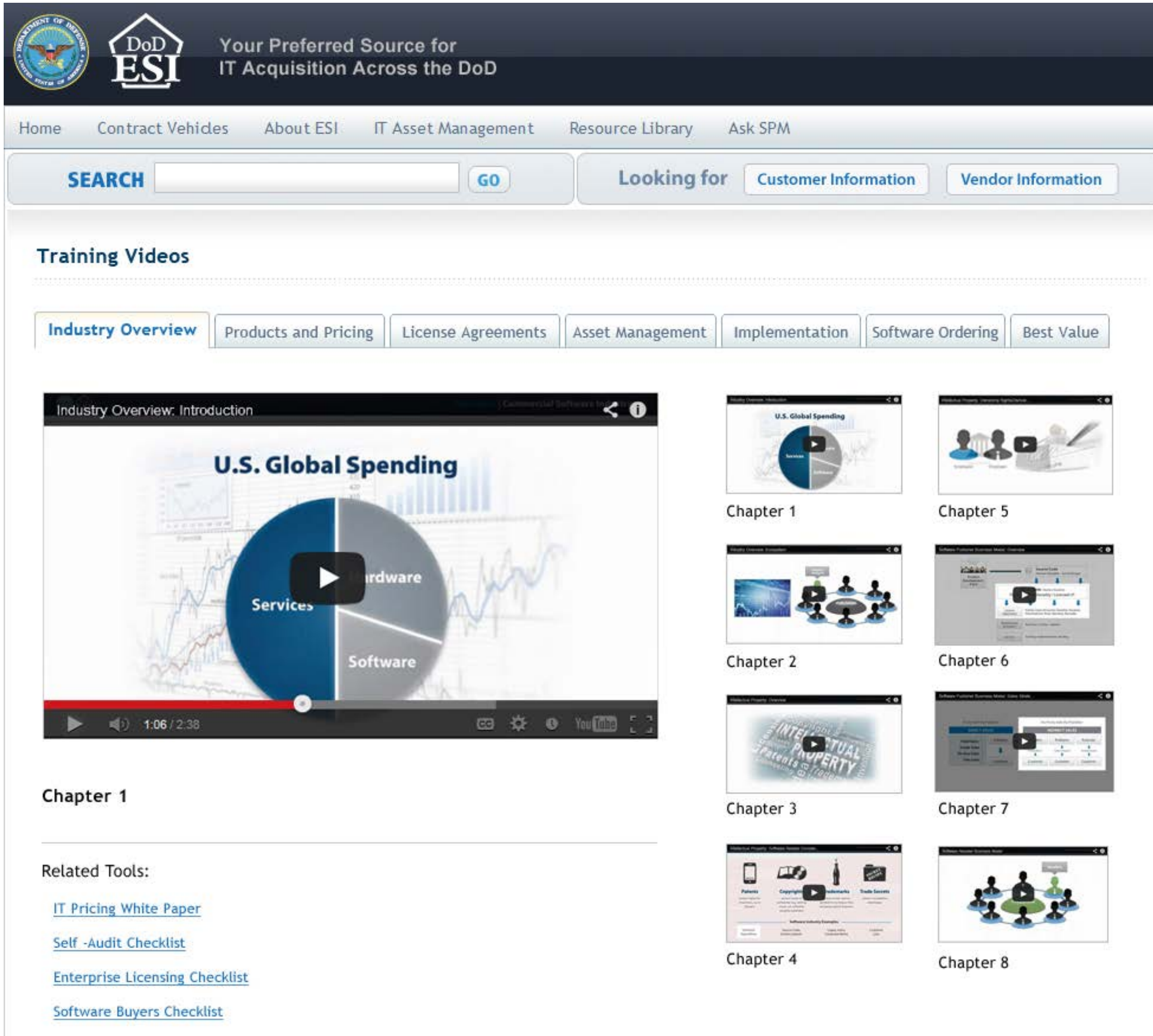
DoD ESI Commercial Software License Acquisition

Throughout the government, program teams need assistance and expertise negotiating software license agreements. This two day session offers training and tools to review and negotiate software license and maintenance agreements and End User License Agreements (EULA). The participants of this session will acquire the initial skills needed to resolve conflicting terms and conditions / licensing issues and achieve effective negotiations skills to support the government's mission.

The first day provides a broad overview of the IT industry and general concepts necessary for license negotiations. The second day provides a much greater level of detail and concentration on a topic selected by the DoD or component training leadership in advance.

Group / Segments	Content
Classes are tailored to fit with a person's experience level and area of responsibility.	Course content is modular so material can be selected for the unique needs of a class or an individual.
Basic Level Courses	
• Anyone who is new to Commercial Software Licensing • Purchase Card Holders • Anyone who wants or needs exposure to commercial software licensing best practices or ESI	• Industry Overview • Publisher Business Model • Reseller Business Model • Intellectual Property • Derivative Works • Products and Services • Pricing • Contracts / Agreements • Market Research See descriptions below.
Specialized / Advanced Courses / Workshops	

DoD ESI Tools: eLearning Tutorials



Up to 8 Modules per Chapter

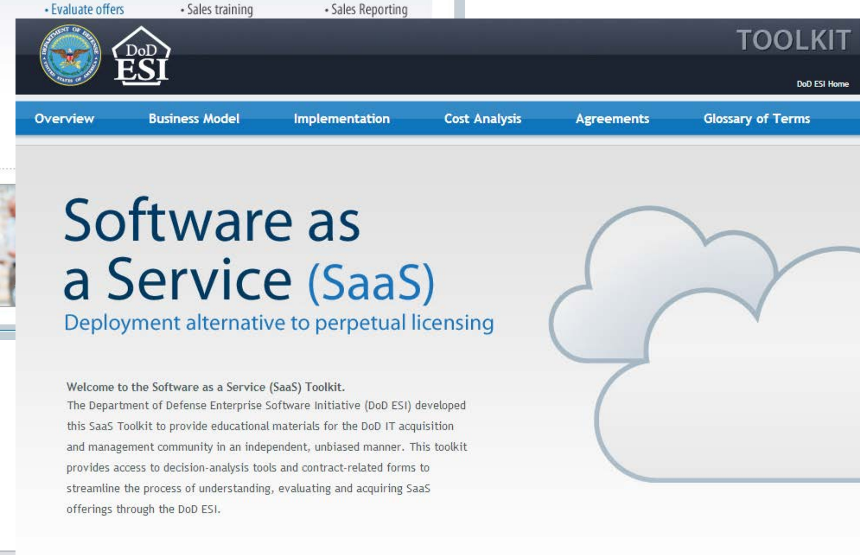
- Industry Overview
- Ordering
- Best Value
- EULAs
- Cloud/SaaS
- Third Party/Open Source
- Virtualization
- Market Research
- Requirements

DoD ESI Tools: HTML Toolkits and Software Buyer's Checklist

Best Value Toolkit

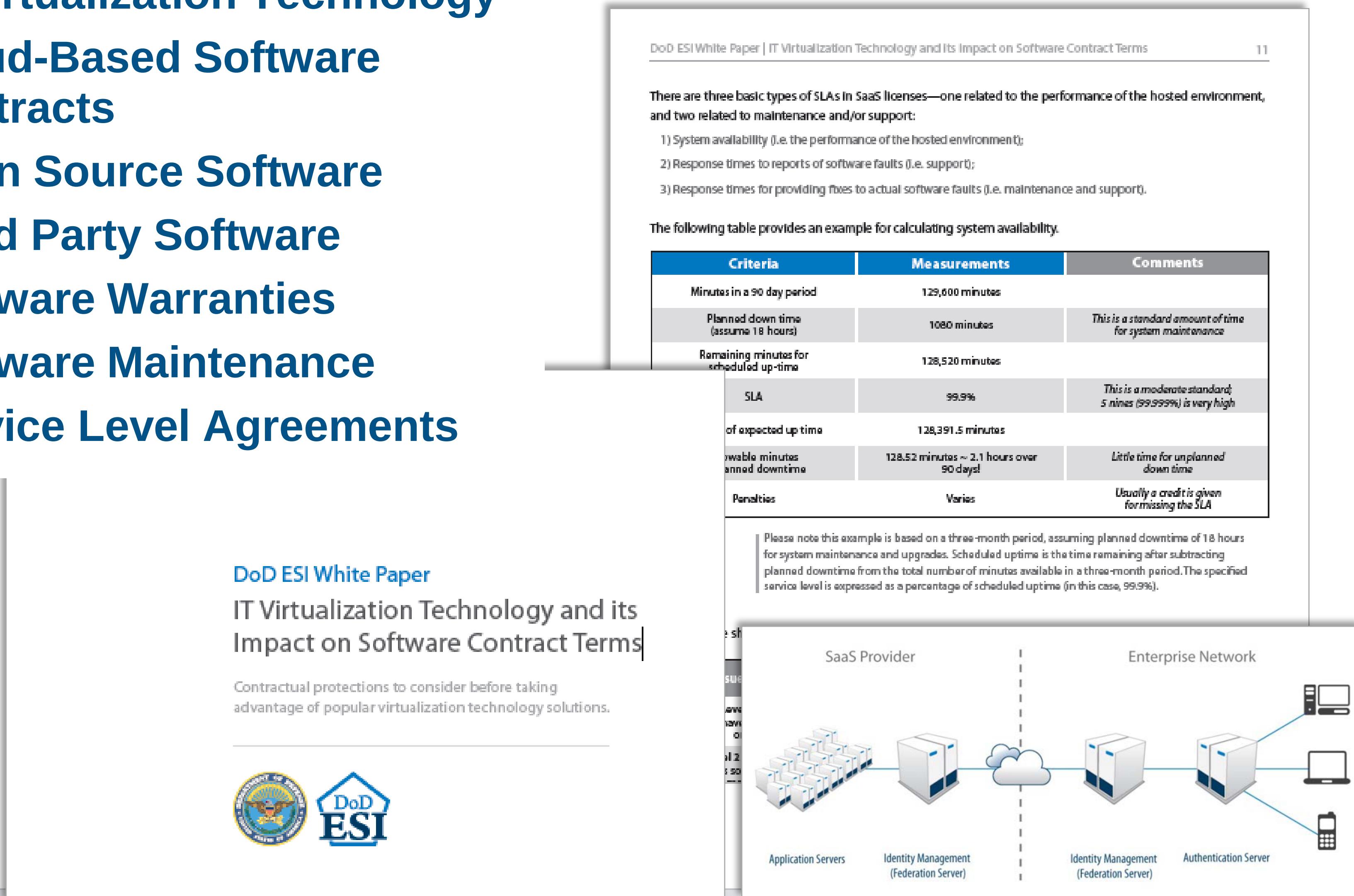
BPA Toolkit

SaaS Toolkit



DoD ESI Tools: White Papers

IT Virtualization Technology
Cloud-Based Software Contracts
Open Source Software
Third Party Software
Software Warranties
Software Maintenance
Service Level Agreements


DoD ESI White Paper | IT Virtualization Technology and Its Impact on Software Contract Terms 11

There are three basic types of SLAs in SaaS licenses—one related to the performance of the hosted environment, and two related to maintenance and/or support:

1) System availability (i.e. the performance of the hosted environment);

2) Response times to reports of software faults (i.e. support);

3) Response times for providing fixes to actual software faults (i.e. maintenance and support).

The following table provides an example for calculating system availability.

Criteria	Measurements	Comments
Minutes in a 90 day period	129,600 minutes	
Planned down time (assume 18 hours)	1080 minutes	*This is a standard amount of time for system maintenance*
Remaining minutes for scheduled up-time	128,520 minutes	
SLA	99.9%	*This is a moderate standard; 5 nines (99.999%) is very high*
…of expected up time	128,391.5 minutes	
…wable minutes …nned downtime	128.52 minutes ~ 2.1 hours over 90 days!	*Little time for unplanned down time*
Penalties	Varies	*Usually a credit is given for missing the SLA*

Please note this example is based on a three-month period, assuming planned downtime of 18 hours for system maintenance and upgrades. Scheduled uptime is the time remaining after subtracting planned downtime from the total number of minutes available in a three-month period. The specified service level is expressed as a percentage of scheduled uptime (in this case, 99.9%).

DoD ESI White Paper

IT Virtualization Technology and its Impact on Software Contract Terms

Contractual protections to consider before taking advantage of popular virtualization technology solutions.

Questions

THE SOFTWARE METRICS CUBE: SOFTWARE ACQUISITION SIMPLIFIED FOR GOVERNMENT OVERSIGHT OFFICES

Mr. Victor Fuster

QSM®

Quantitative Software Management, Inc.

"Simplicity is the ultimate sophistication."

- Commonly attributed to Leonardo da Vinci

Disclaimer

- Many published approaches to software metrics identification and methodology, software costing, etc. (I've leveraged many!)
- This presentation is based on implementation experiences in the government oversight office environment
  - Produced measurable results
  - Gained recognition from senior leadership

Overview

- The Oversight Organizational Environment
- Why Simplify?
- Identify Measures/Metrics
- Collect Data
- The Software Metrics Cube

Oversight Organizations

- Principal Advisor to…based on areas of expertise
- Primary objective is not actively managing software activities
- Provide the information and insight to support even higher senior-level decision making

Oversight Organizations

- Pressure to demonstrate the value of their insight at decision points and collectively across an organization
- Predictability in a world of uncertainty
  - Positive uncertainty is opportunity
  - Negative uncertainty is risk

Oversight Organizations

- Oversight priorities may be to:
  - Assess project/program/portfolio plans for feasibility (schedule, cost, etc.)
  - Assess project/program/portfolio health at a snapshot in time (milestones/gates/breach)
  - Identify trends and conduct comparisons of larger groups
  - Provide policy and guidance

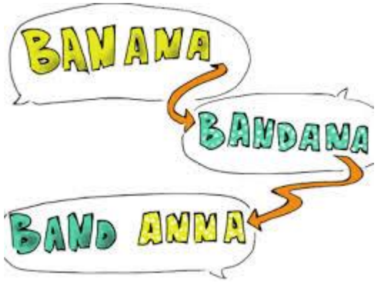
"The nine most terrifying words in the English language are, 'I'm from the government and I'm here to help."

- President Ronald Reagan

Oversight Organizations

Challenges (as they relate to software acquisition management and software estimation)

- Detached from the software development activities, which often causes challenges in the ability to fully leverage data:
  - information collection process
  - information context
  - information quality



- Lack software domain experience

Oversight Organizations

Despite these challenges…want to:

- Harness the power of "all the data we have"
- Provide quantifiable findings through use of that data ☺

The goal is to turn data into information, and information into insight.

-Carly Fiorina, Former CEO of HP

Let's Use Data!

Why Simplify?

QSM®
Quantitative Software Management

Why simplify?

Big Picture

- Getting fastest ROI / Proof of concept
- Get past analysis paralysis
- Culture change / leadership change

Why simplify?

Where the rubber meets the road…

- Focus limited resources on core set of metrics that can provide trends we can leverage for future estimates
- Gain larger, more complete data samples
- Limit situations where we can only base estimates off just one or two analogous projects

Identify Your Measures/Metrics

…a first step to the cube

Identify Your Measures/Metrics

- Use a methodology to identify the important information needs for your decision makers

  For example only:

  - Goal-Question-Metric (GQM)

- Literature review of current DoD and relevant Industry guidance

  For example only:

  - Software Resources Data Report (SRDR) (2015)
  - Software Cost Estimation Metrics Manual for Defense Systems (2015)
  - Supplement to Guidebook for Naval Software Intensive Systems (2010)
  - USAF Weapon Systems Software Management Guidebook (2008)

Example Measures

- Size/scope
- Duration/Schedule
- Productivity
- Effort/Staffing
- Defects/Reliability
- Cost

People, working at some level of *productivity*, produce a quantity of function or a *work product* at a level of *reliability* by the expenditure of *effort* over a *time* interval.

- Larry H. Putnam

"Size" Example

- **A proxy for the value and knowledge content of the delivered system—what the system is worth**
- **Size can be indicated by a number of metrics:**

<u>Front end: Unit of Need</u>

Based on characteristics of the statement of needs

- Requirements
- Function Points/Object Points
- IO Counts
- States/Events/Actions
- Use Cases
- Stories/Story Points
- Objects/Classes
- Components
- Web Pages

<u>Back end: Unit of Work</u>

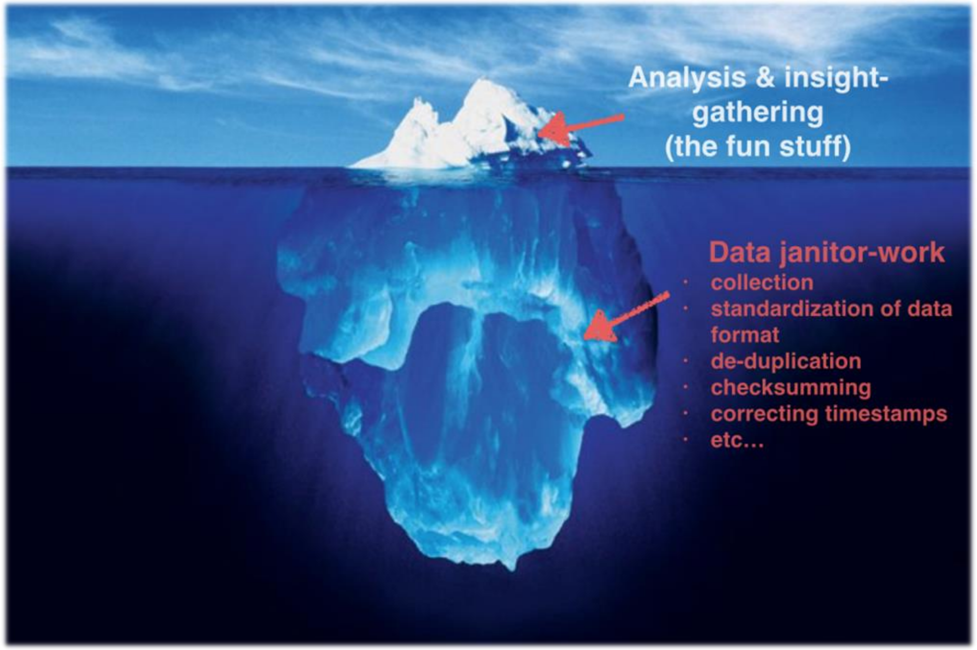
Based on the characteristics of the system when built

- Lines of Code
- Statements
- Actions
- Modules
- Subsystems
- GUI Components
- Logic Components
- Logic Gates
- Tables

Each unit has a "relative weight" and precision.
Front end units tend to be less precisely <u>defined</u>, while the prediction of the <u>count</u> of back end units tends to be less precise.
The relative weight measures the size or "complexity" of the *unit* and is called a Gearing Factor.

Collect Data

…a second step to the cube



"Data scientists, according to interviews and expert estimates, spend from 50 percent to 80 percent of their time mired in this more mundane labor of collecting and preparing unruly digital data, before it can be explored for useful nuggets."

- Steve Lohr, New York Times, 2014

Templates

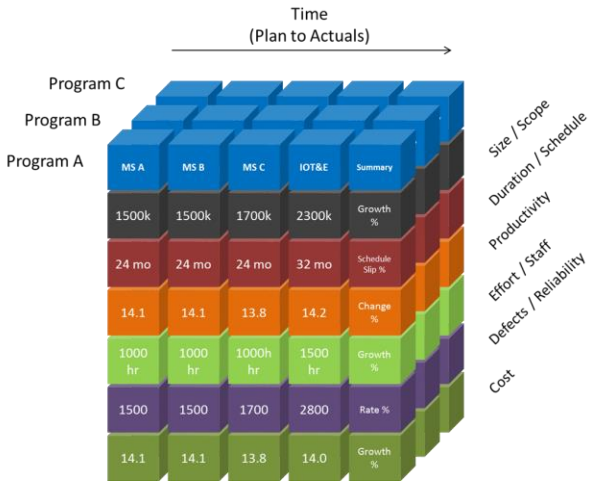
- We use focused templates that align to our simplified measures
- Recognize human and organizational behavior
- Force multiplier in data collection

Templates

- Size/scope
- Duration/Schedule
- Productivity
- Effort/Staffing
- Defects/Reliability
- Cost

Category	Baseline Plan	Current Plan	Actual	Details	Rationale	Source
Measure	As-of date	As-of date	As-of date	*(See Data Dictionary tab for additional guidance)*		
Size						
New (Units)	#	#	#	SW newly written for this system release.	New SW size provides a means to normalize other measures. Plan vs. Actual provides a means to conduct variance analysis.	
Schedule/ Duration						
Phase Start Date	date	date	date	Start Date of Phase: During this phase, cost and technical feasibility are established and very high level software requirements are defined.	Phase start date bounds all effort, SLOC and duration for a given phase. Comparing baseline start date to actual start date is an indicator of the "lateness" of this phase.	
Staffing/ Effort						
Peak SW FTE	#	#	#	Head count or effort hours	Provides an indication of the size of the "envelope" of effort for a given system release. Useful for parametric estimation and forecasting.	
Defects/ Reliability						
Defects Discovered			#	Defects discovered by month by severity[7](with date discovered for each defect), provide data in format of choice	Relative trajectory over time provides a measure of SW maturity, and readiness for down-stream events (e.g., DT, customer delivery)	
Productivity						
Productivity				Derived from size, time, effort data	Indicates the efficiency with which the SW team is able to delivery required functionality. Instrumental in predicting future SW developments (e.g., effort, duration, size, quality).	

Software Metrics Cube



…we have arrived

QSM®
Quantitative Software Management

Software Metrics Cube

- A simplified measurement approach to enable more efficient use of available software development/sustainment metrics
- Supports:
  - Prediction/estimation of performance, risk, cost, etc.
  - Analytical calculations (statistical modeling) that answer “what if?” questions to provide better courses of action
  - Benchmarking that enables process improvement and competitive analysis

Decision-makers want predictability!

Other information in hand…

- With appropriate caveat*, can provide valuable snapshots of an organization's data

  *Senior leaders often have little time to learn the many complex methodologies they encounter, so they may gravitate to "snapshots"

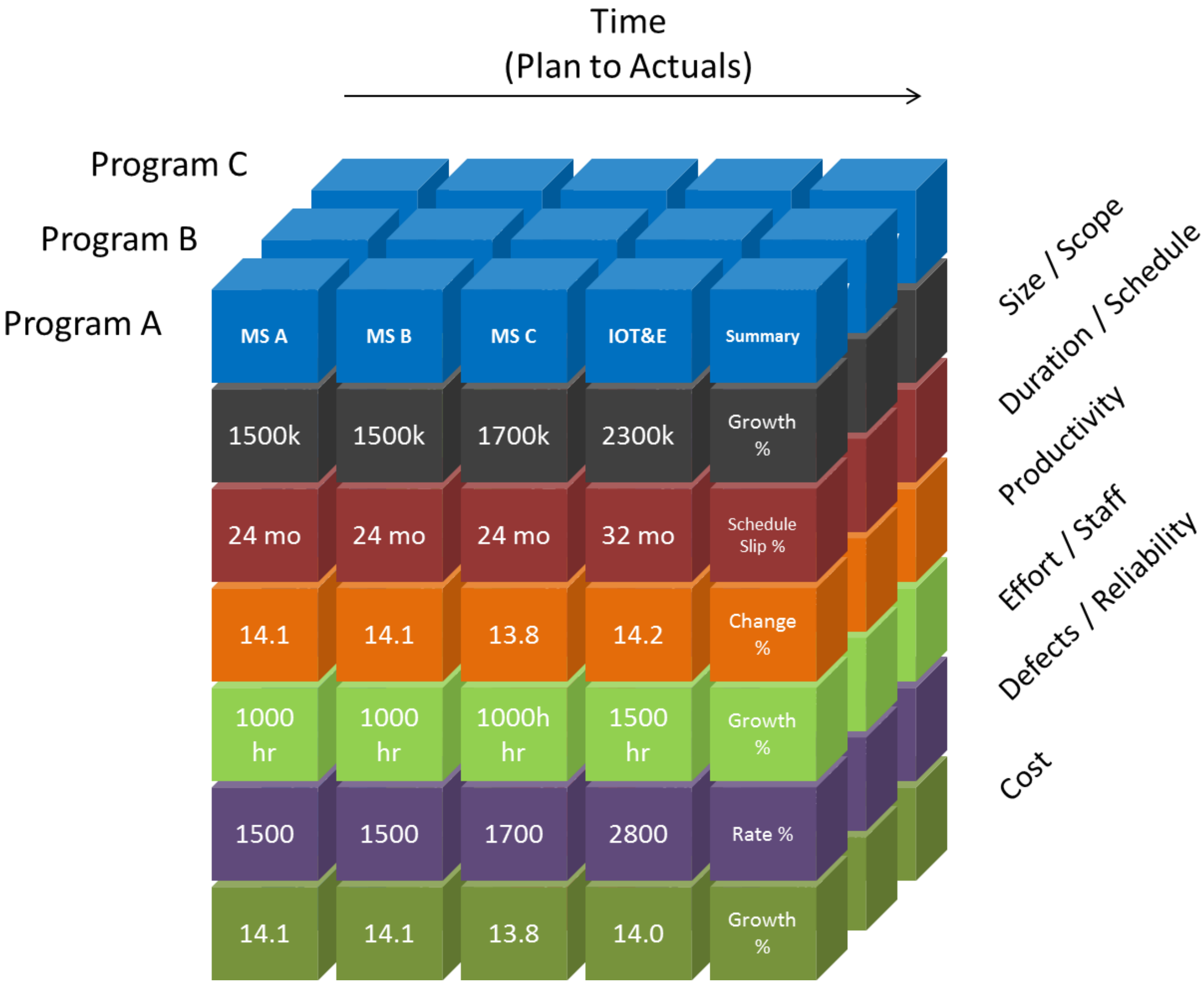
"…based on our data, we've observed that a SW project with X functionality, being developed by agile methods, in X DoD domain ranges in cost from X to X…

Software Metrics Cube

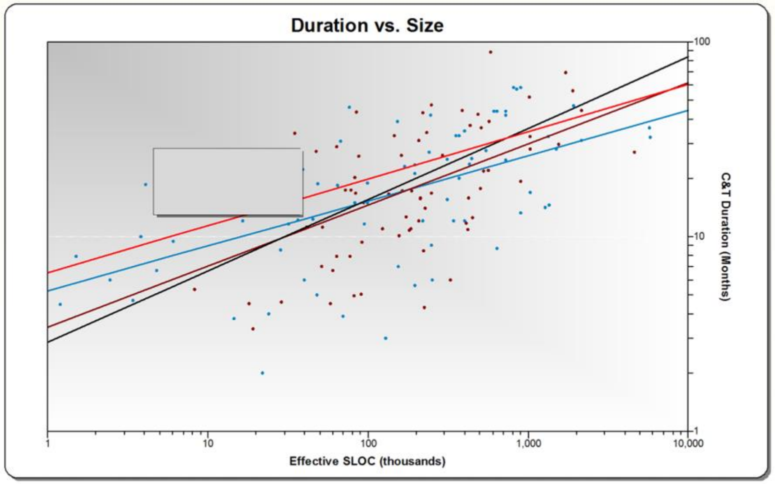
	Size/scope	Duration/schedule	Productivity	Effort/Staff	Defects	Cost	Risk (cost/schedule/ performance)
	Summary	Summary	Summary	Summary	Summary	Summary	Summary
Program A	1500k	24 mo	24 p	3300 ph	44.0	4mil	Program A
Program B	1000k	18 mo	24 p	7200 ph	15.0	3mil	Program B
Program C	2300k	40 mo	29 p	21000 ph	6.7	5mil	Program C
Program D	500k	12 mo	8 p	2900 ph	27.4	450k	Program D

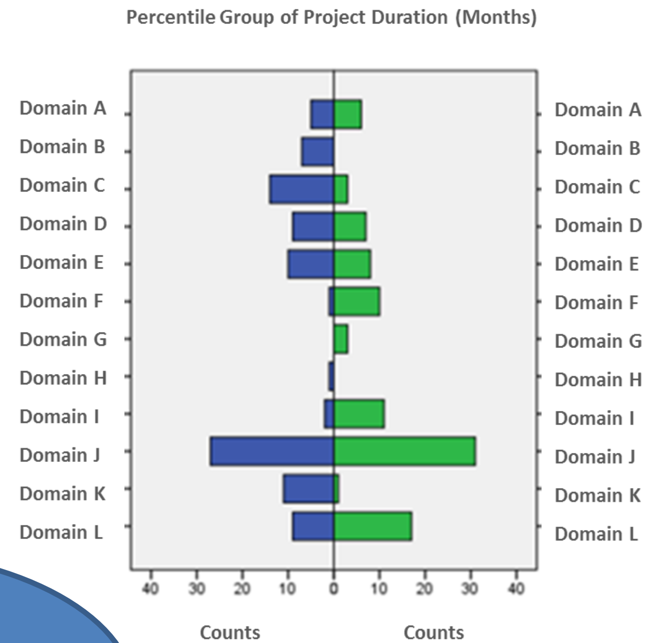
QSM®
Quantitative Software Management

Software Metrics Cube

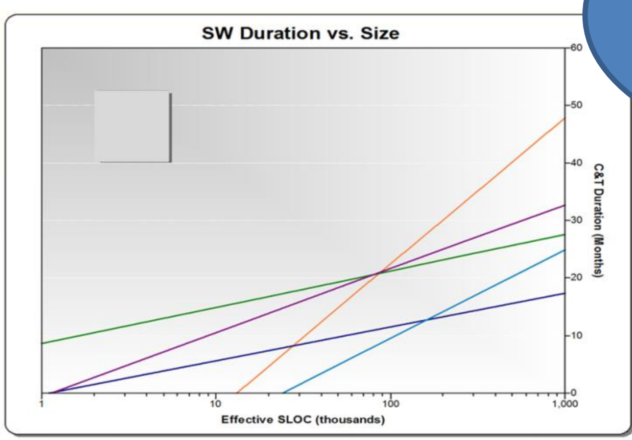


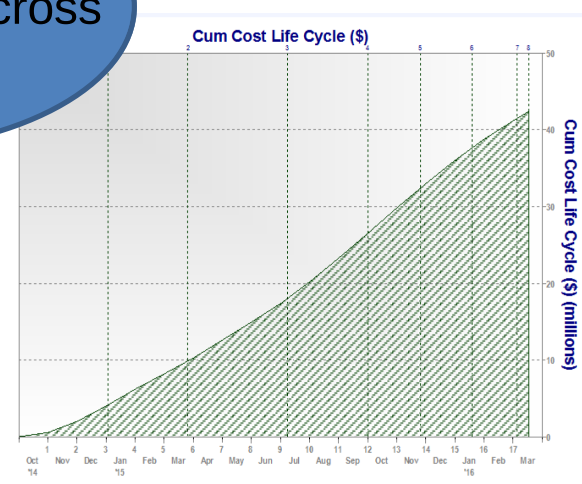
Benchmark Examples





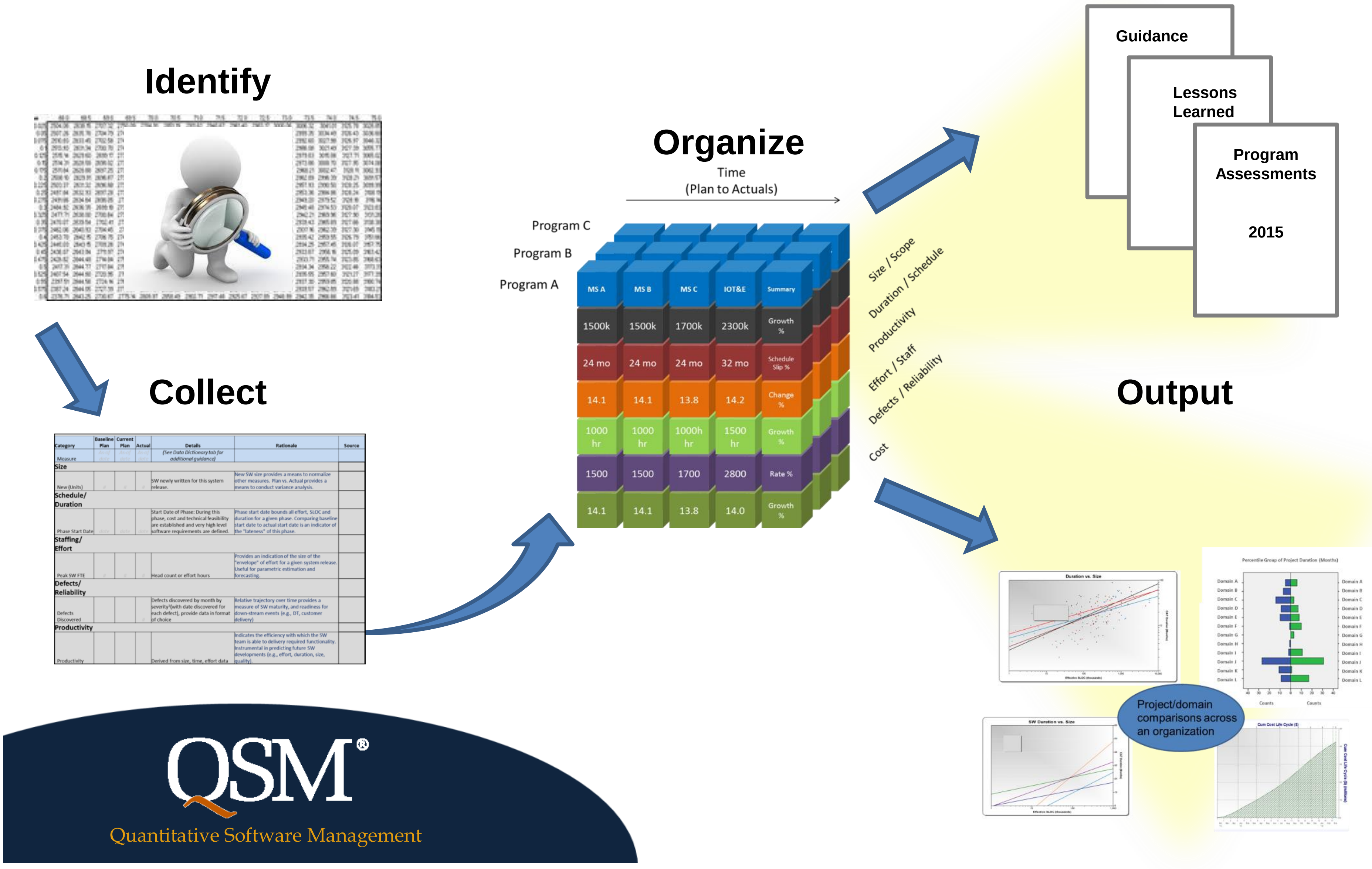
Project/domain comparisons across an organization





The Intelligence behind
Successful Software Projects

Software Metrics Cube



Conclusion

- Oversight organizations that are responsible for software acquisition management require a quantitative approach that demonstrates ROI to the senior-decision makers they support

- Software Metrics Cube is a simplified measurement approach to enable more efficient use of available software development/sustainment metrics

  - Utilizes simplified Identification, Collection and Organization of software measures that support analysis and quality products

Questions?

estimate

estimate • analyze • plan • control

Why Can't People Estimate: Estimation Bias and Mitigation

Dan Galorath

galorath@galorath.com

310 414-3222 x614

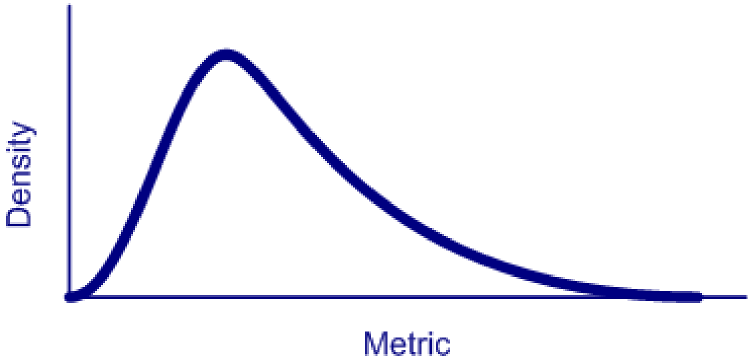
SEER by GALORATH

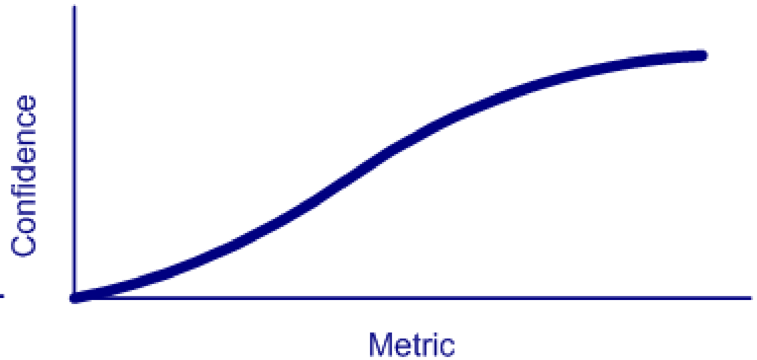
ESTIMATION & PLANNING:

An Estimate Defined

- An ***estimate*** is the most knowledgeable statement you can make ***at a particular point in time*** regarding:
  - Effort / Cost
  - Schedule
  - Staffing
  - Risk
  - Reliability

- Estimates more precise with progress
- *A WELL FORMED ESTIMATE IS A DISTRIBUTION*





Estimation Methods Summarized

Category	Description	Advantages	Limitations
Guessing	Off the cuff estimates	Quick Can obtain any answer desired	No Basis or substantiation No Process Usually Wrong
Analogy	Compare project with past similar projects.	Estimates are based on actual experience.	Truly similar projects must exist Or analogy techniques used
Expert Judgment	Consult with one or more experts.	Little or no historical data is needed; good for new or unique projects.	Experts tend to be biased; knowledge level is sometimes questionable; may not be consistent.
Vendor Quotes	Vendor identification of scope & costs	Vendor has experience and (hopefully) data Vendor can commit to scope	Often assume best case.. Then exceed Customer costs not included
Agile Velocity		Helps root level management of Agile Projects	Doesn't estimate up-front well or provide answers for management decision making
Comprehensive Parametric Models	Perform overall estimate using design parameters and mathematical algorithms.	Models are usually fast and easy to use, and useful early in a program; they are also objective and repeatable.	Models can be inaccurate if not properly calibrated and validated; Bias in parameters may lead to underestimation.

Human Nature:
Humans Are Optimists

Harvard Business Review explains this Phenomenon:

- Humans seem hardwired to be optimists
- Routinely exaggerate benefits and discount costs

Delusions of Success: How Optimism Undermines Executives' Decisions (Source: HBR Articles | Dan Lovallo, Daniel Kahneman | Jul 01, 2003)

Solution - Temper with "outside view":
Past Measurement Results, traditional forecasting, risk analysis and statistical parametrics can help

Don't remove optimism, but balance optimism and realism

Cognitive Bias: How Fair Are We
(Source BeingHuman.org)

- Cognitive bias: Tendency to make systematic decisions based on cognitive factors rather than evidence
- Human beings exhibit inherent errors in thinking
- Researchers theorize in the past, biases helped survival
  - Our brains using shortcuts (heuristics) that sometimes provide irrational conclusions

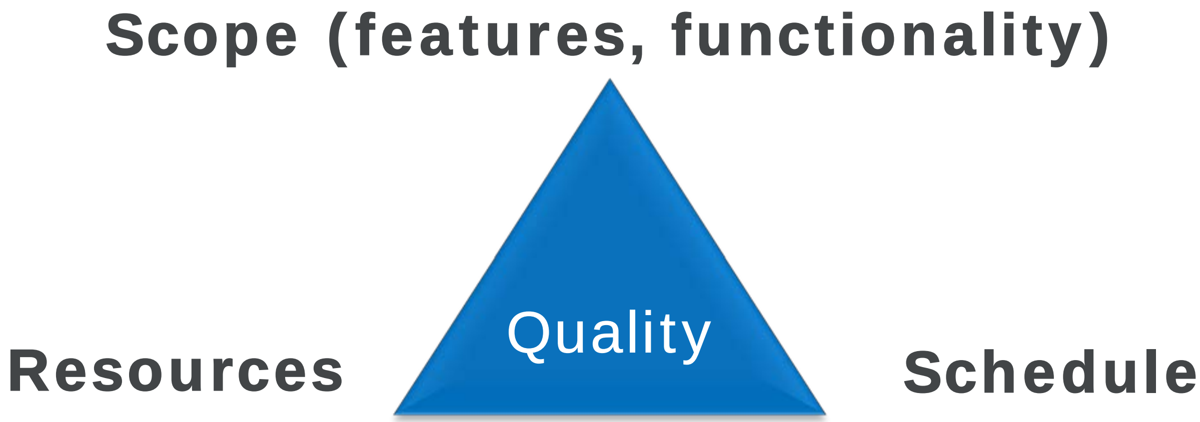
"We usually think of ourselves as sitting the driver's seat, with ultimate control over the decisions we made and the direction our life takes; but, alas, **this perception has more to do with our desires—with how we want to view ourselves—than with reality**." Behavioral economist Dan Ariely

- Bias affects everything:
  - from deciding how to handle our money
  - to relating to other people
  - to how we form memories

Essence of the problem: Memory is unreliable and we are hard wired to ignore risk & questioning

Trouble Starts By Bias or Strategic Mis-Estimation Ignoring Iron Triangle

- Typical Trouble: Mandated features needed within specific time by given resources



- At least one must vary otherwise quality suffers and system may enter impossible zone!

Sometimes strategic mis-estimation
is used to get projects started or to win
Some customers think price to win is strategic mis-estimation (it is not)

The Planning Fallacy (Kahneman & Tversky, 1979)

- Judgment errors **are systematic & predictable**, not random
  - Manifesting bias rather than confusion
  - Judgment errors made by experts and laypeople alike
  - Errors continue when estimators aware of their nature
- Optimistic due to overconfidence ignoring uncertainty
  - Underestimate costs, schedule, risks
  - Overestimate benefits of the same actions
- Root cause: Each new venture viewed as unique
  - "inside view" focusing on components rather than outcomes of similar completed actions
  - FACT: Typically past more similar assumed
    - even ventures may appear entirely different

Reference Class Forecasting (adapted from http://www.slideshare.net/assocpm/a-masterclass-in-risk)

- Best predictor of performance is actual performance of implemented comparable projects (Nobel Prize Economics 2002)
- Provide an "outside view" focus on outcomes of analogous projects
  - Attempts to force the outside view and eliminate optimism and misrepresentation
- Choose relevant "reference class" completed analogous projects
- Compute probability distribution
- Compare range of new projects to completed projects

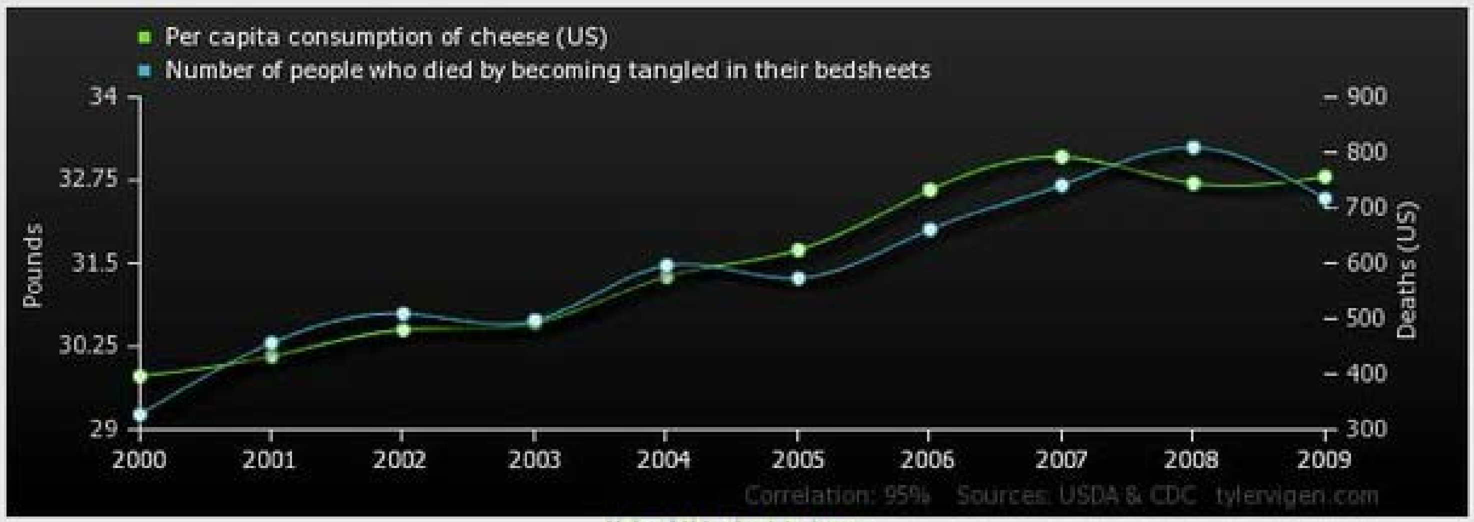
Correlation Doesn't Always Mean Causation (Source: www.memolition.com)

SEER by GALORATH

Per capita consumption of cheese (US)

correlates with

Number of people who died by becoming tangled in their bedsheets



Upload this chart to imgur

	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
Per capita consumption of cheese (US) Pounds (USDA)	29.8	30.1	30.5	30.6	31.3	31.7	32.6	33.1	32.7	32.8
Number of people who died by becoming tangled in their bedsheets Deaths (US) (CDC)	327	456	509	497	596	573	661	741	809	717

Correlation: 0.947091

Adding Reality to Estimates – Example – 2 (Source SEI)

Step		Expected	
1		30	
2		50	
3		80	
4		50	
5		90	
6		25	
7		35	
8		45	
9		70	
10		25	
		500	

What would you forecast the schedule duration to be now?

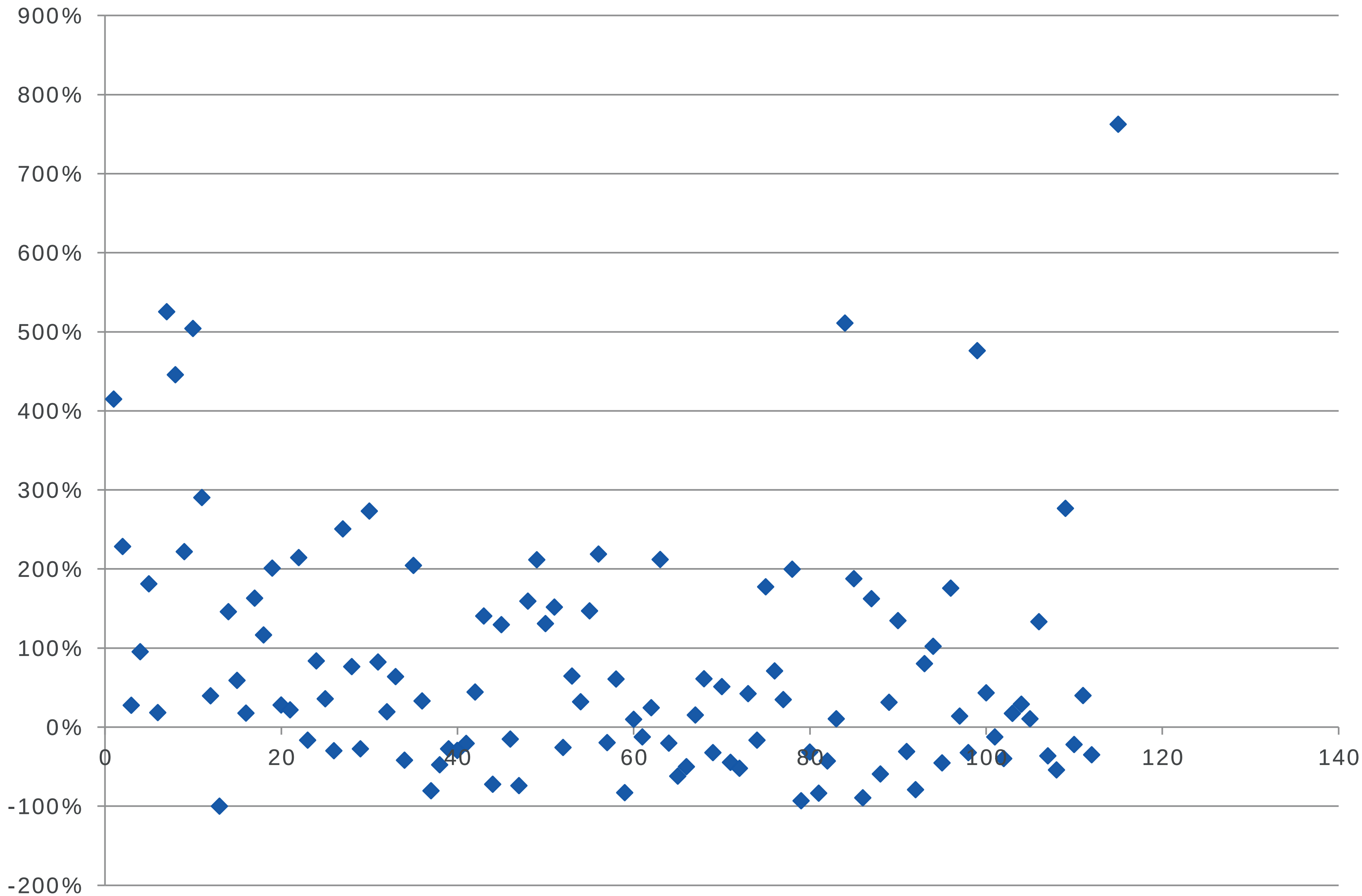
Example Bias Mitigation Using Multiple Sources

Evaluate All Sources of Software Size…

Total Size Estimates	Least	Likely	Most
Expert Judgement	12000	15500	17000
Relevant Range by Analogy	19850	24750	32540
Sizing Database	8000	32000	46000
Functional Analysis	19680	27540	35400
SEER-EstimateByCompare	15450	22650	29850
Delphi Analysis	16788	19750	22713
Estimate Range	**12000**	**22650**	**46000**

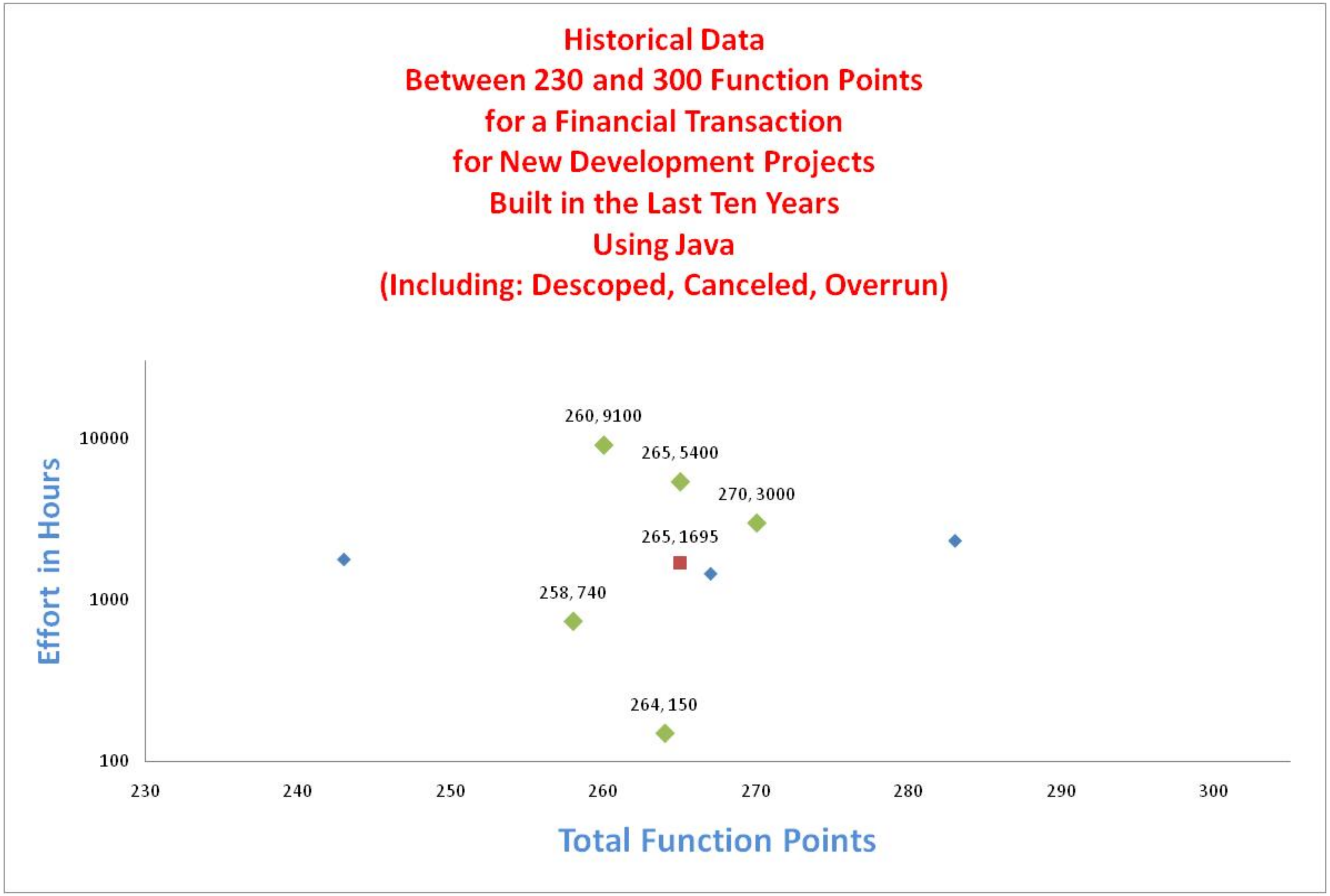
Estimate Independently then show table to minimize anchoring and other bias

SRDR v1 Estimate New SLOC vs Actual (Note: HUGE outliers removed to make the graph more readable)



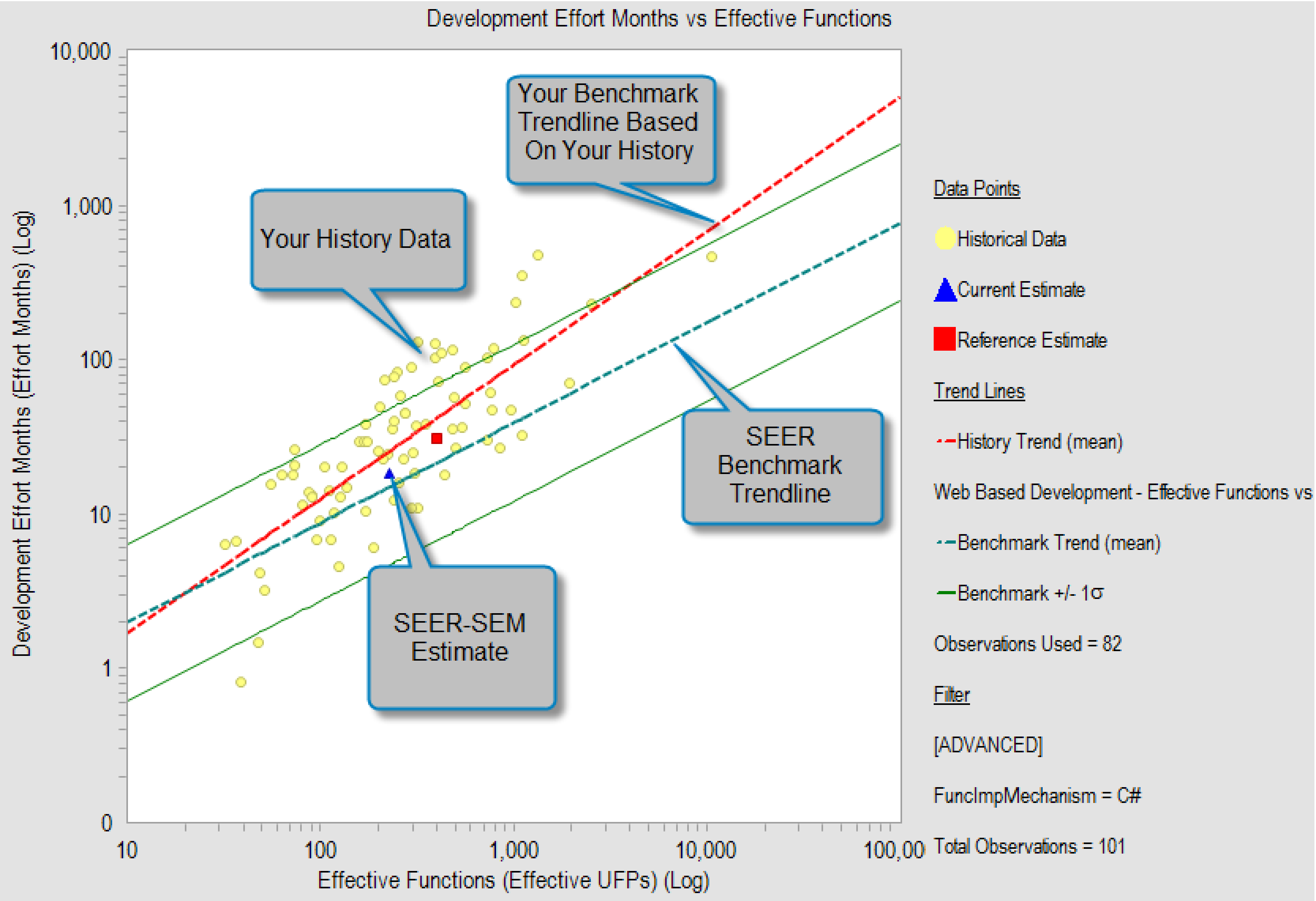
Gross underestimation of software size versus actual

Fallacy of Silent Evidence
What about what we don't know?

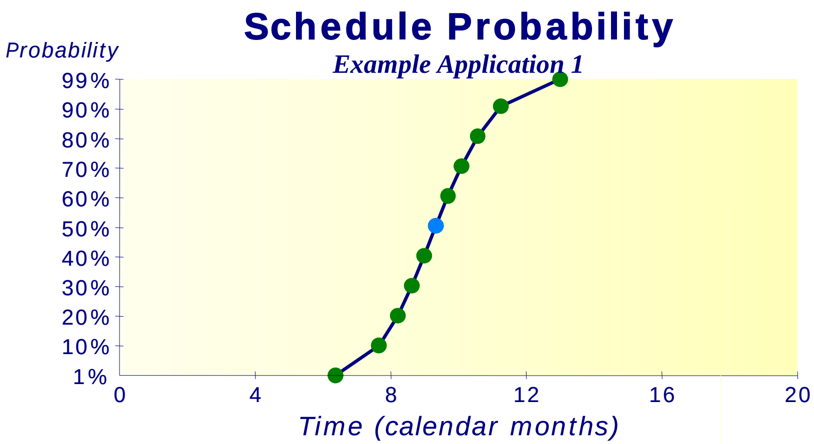


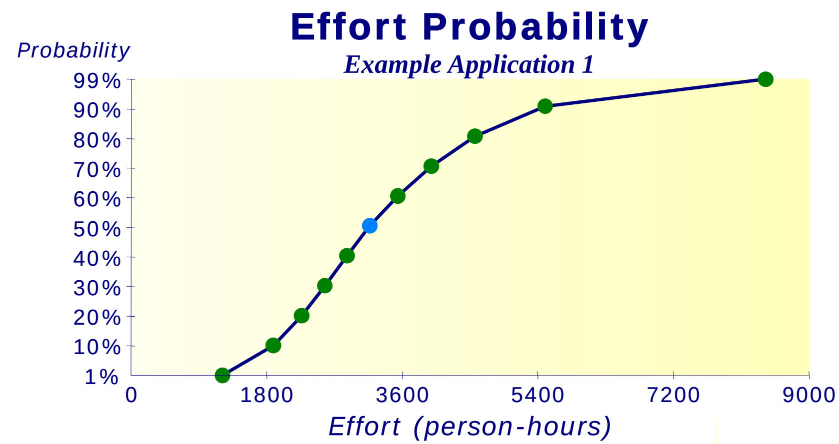
How confident would you feel if the Silent Evidence was visible?

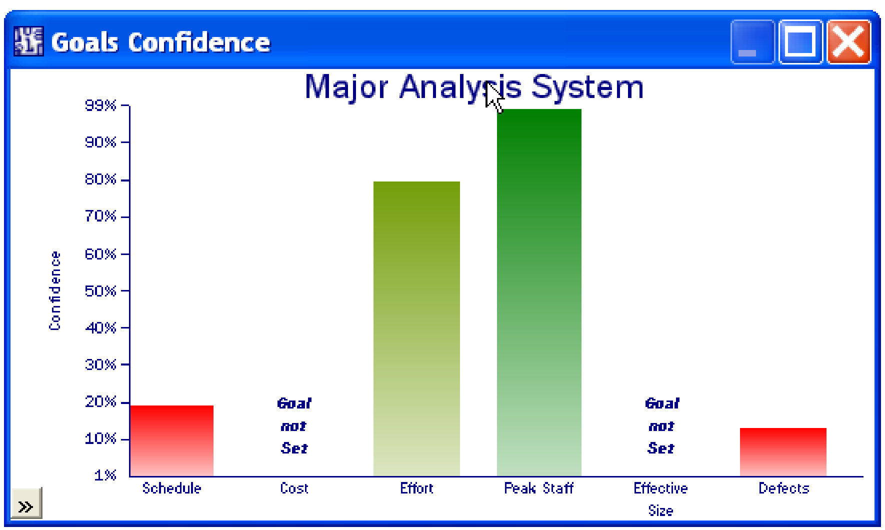
Example: Parametric Estimate Compared With History

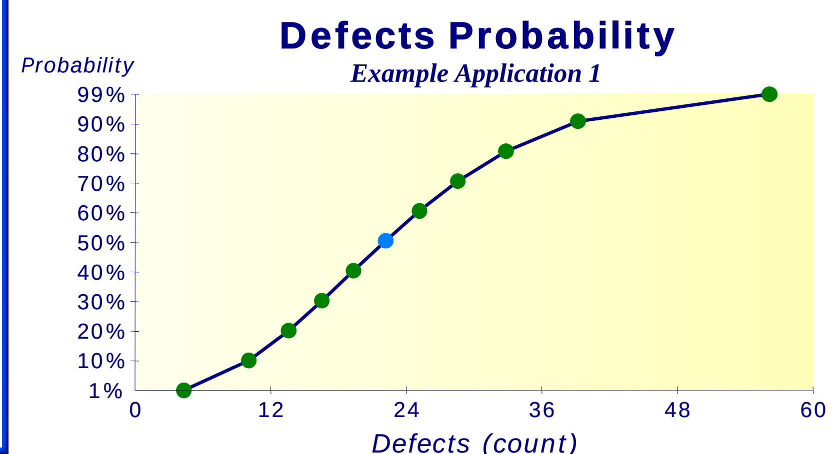


Understand Project Risks Include Them In Planning Decisions (Example SEER-SEM Outputs)



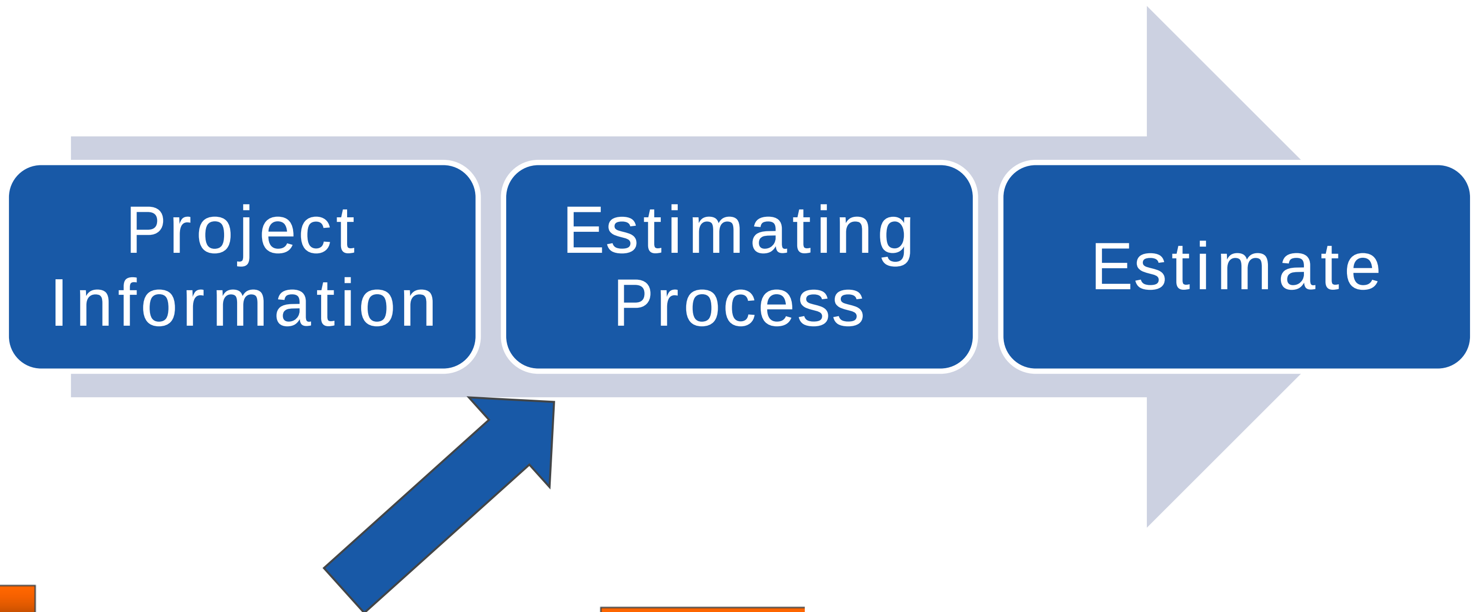


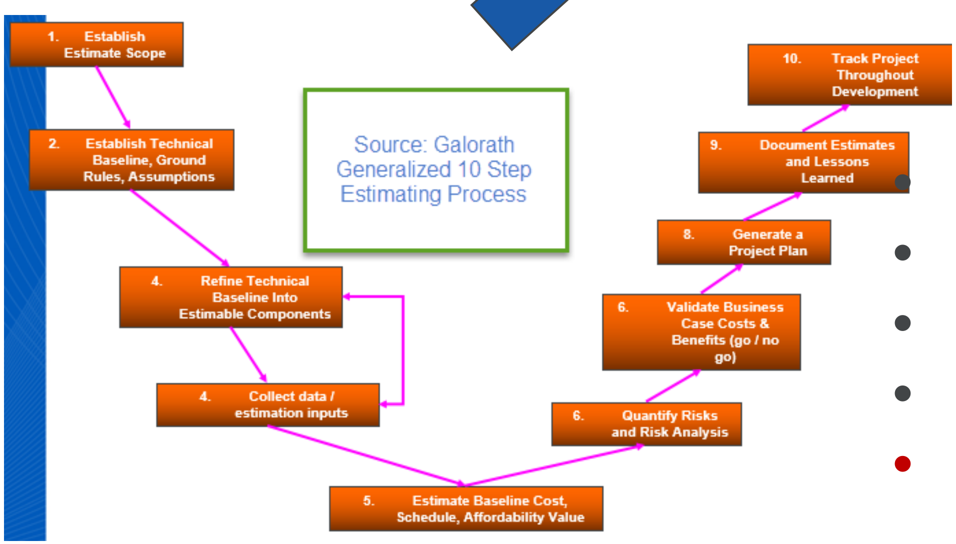




Estimating Process Should Help Mitigate Bias (Adapted from Andy Prince)

SEER by GALORATH





Process Provides

- Traceability
- Repeatability
- Best Practices
- Analytical Mindset
- STEPS TO MITIGATE BIAS

Anchoring Experiment: Anchoring Biases Estimates (Source: myweb.liu.edu/~uroy/eco23psy23/ppt/04-anchoring.pptx)

SEER by GALORATH

1. Subject witnesses the number that comes up when a wheel of fortune is spun
2. Is asked whether the number of African countries in the U.N. is greater than or less than the number on the wheel of fortune
3. Is asked to guess the number of African countries in the U.N.

Result: those who got higher numbers on the wheel of fortune guessed bigger numbers in Step 3

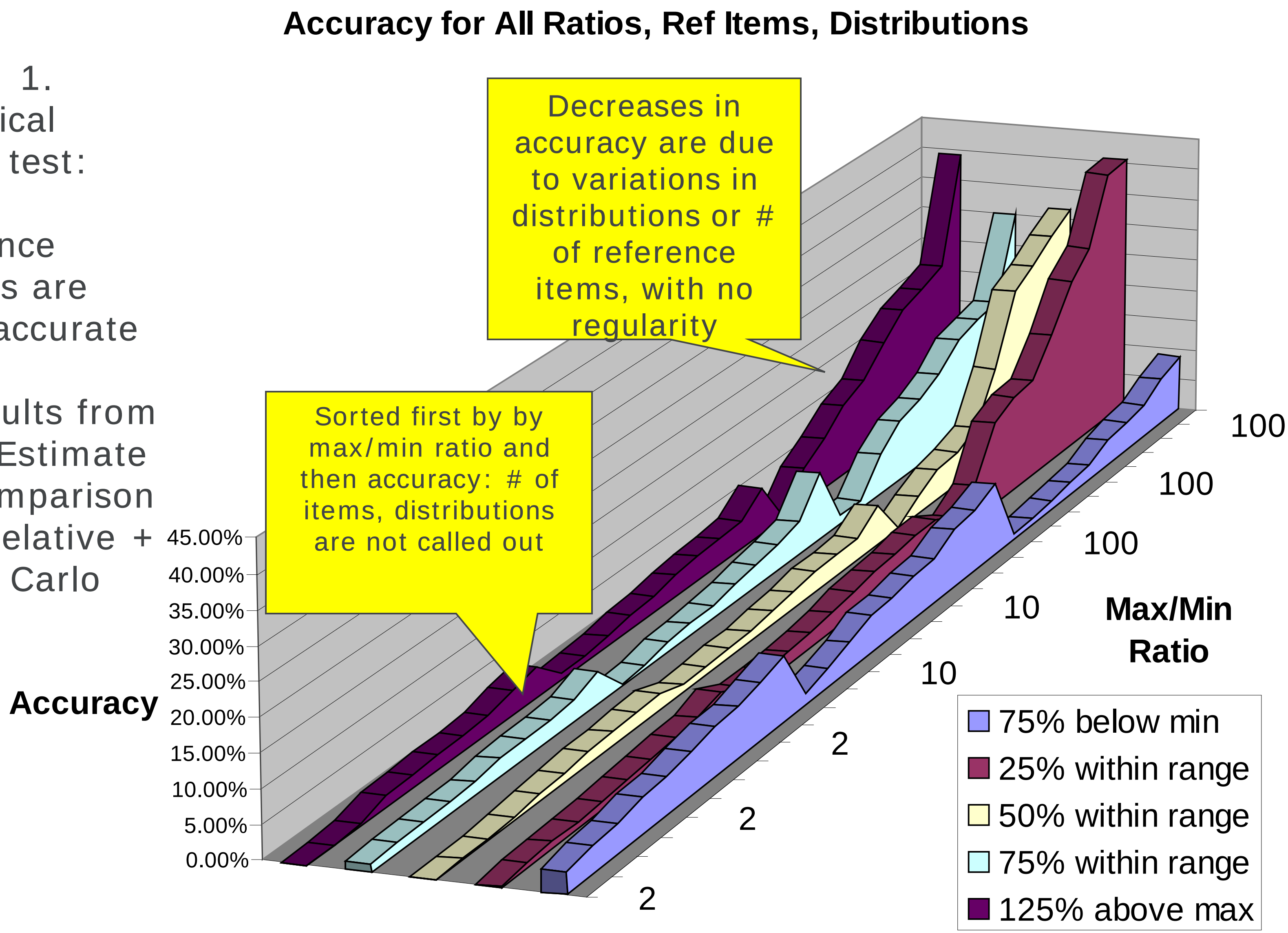
If given a number that biases estimates


Anchoring
How we choose by comparing with a nearby reference point

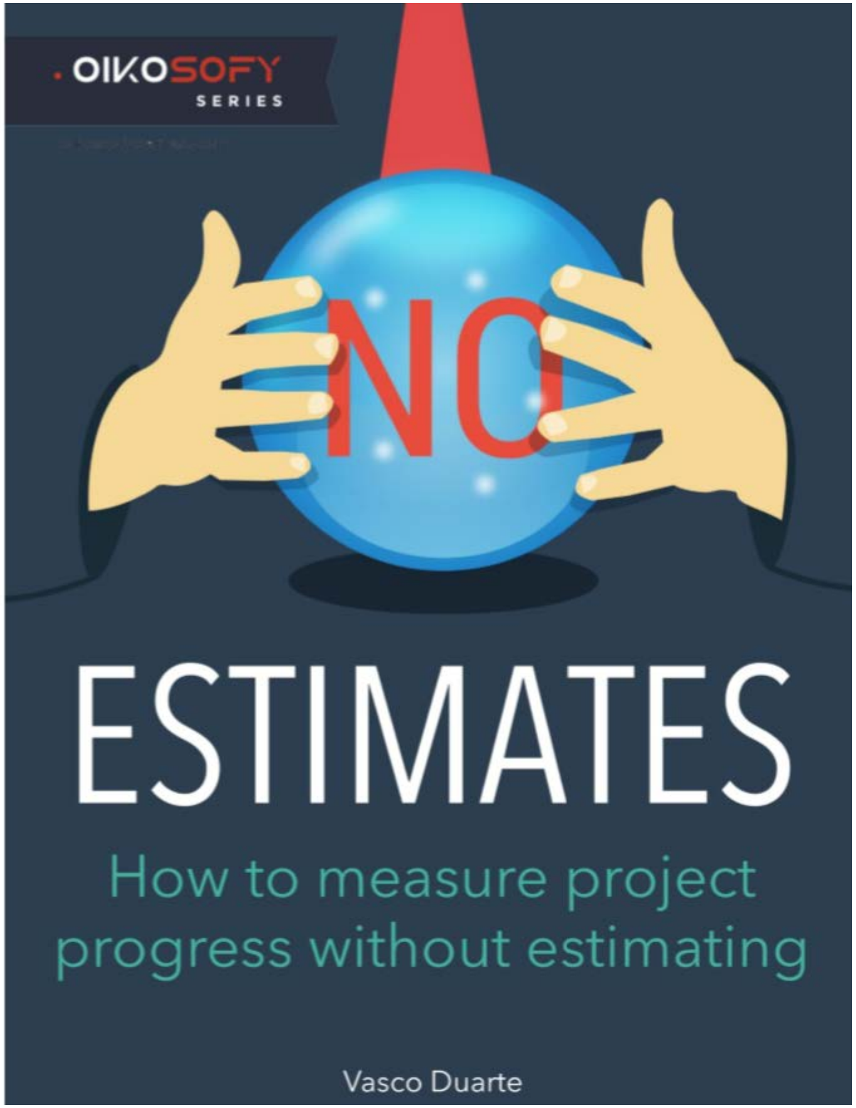
AHP Type Relative Analysis Can Be Within 10% of Actuals

Notes: 1. statistical stress test: Viable reference choices are most accurate

2. Results from SEER Estimate By Comparison Uses relative + Monte Carlo

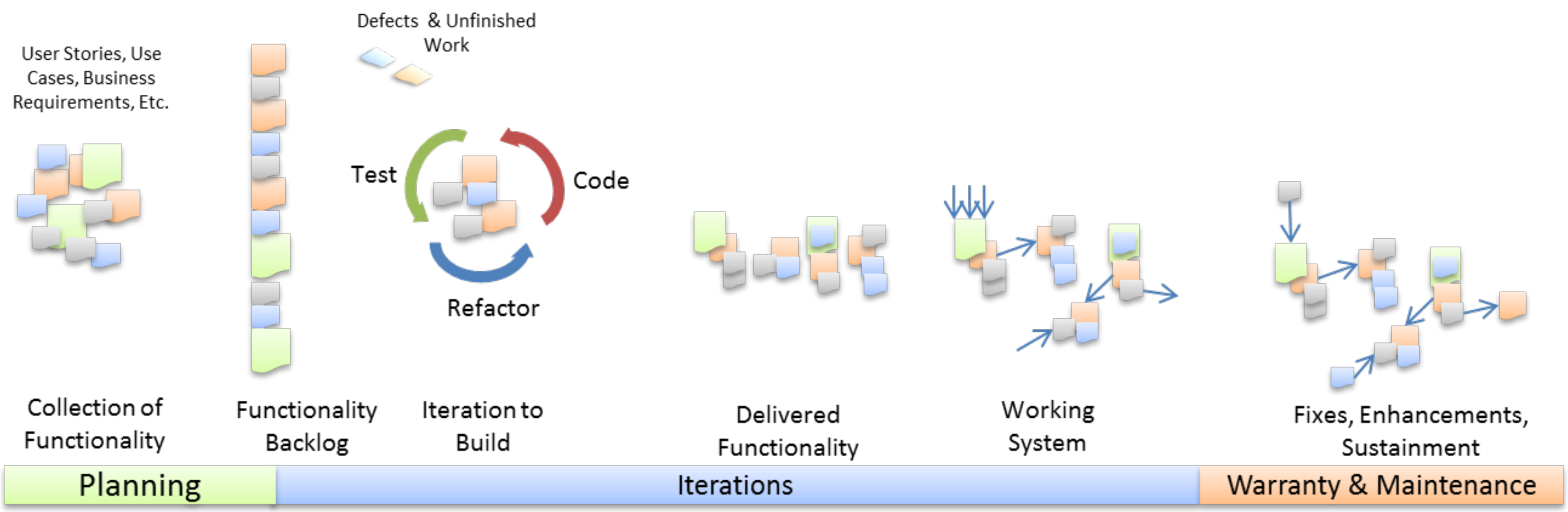


Add In The Agile Bashing of Estimating For a Full View



The Agile "Life Cycle"
(Scrum Example)

- Focus is on what features can be delivered per iteration
- Not fully defined what functionality will be delivered at the end?

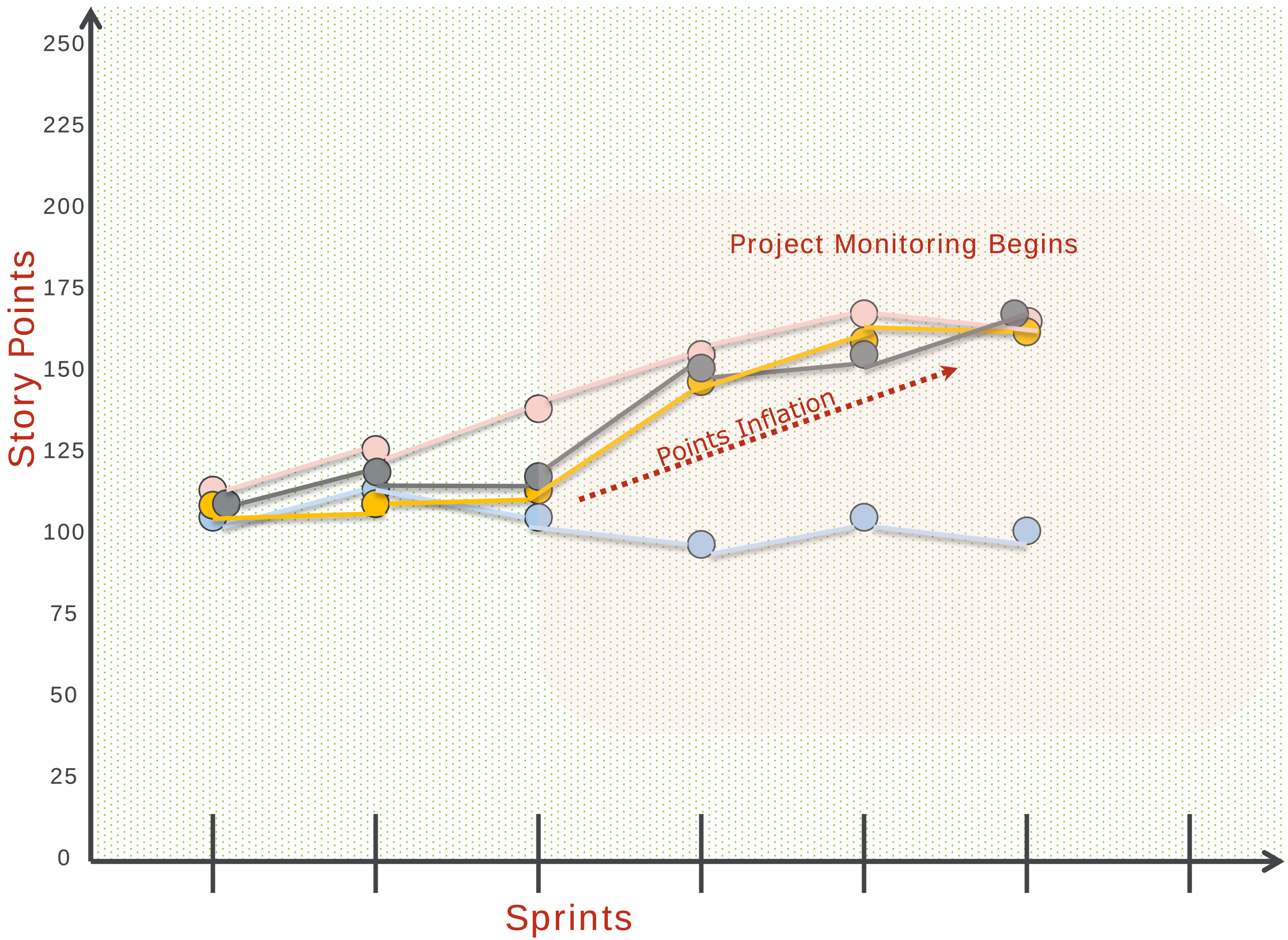


- Iterations are often called a "Sprint"

Root Causes Of Bad Estimates & Bias In Agile Projects As An Example

- Team not really doing Agile
  - Everyone seems to have their own "hybrid" which is code for management controls
- Immature process
  - No one with previous experience, i.e.: no Scrum Master
  - No training in the process being used
- Management gets in the way
  - Micromanage the burn down chart
  - Want to use velocity as productivity
  - Assume Ideal Days = Capacity Days
- Bad Story Counting
  - Trying to use counts across teams
  - Using historical story point counts for new work

Inflation in Story Point Productivity



Key Points

Without care estimates are usually biased (even with experts)

Tempering with an "outside view" can mitigate some bias

Estimates can be better, squelching bias & strategic mis-estimation... Parametrics help.

estimate

estimate • analyze • plan • control

Backup slides

SEER by GALORATH

Confirmation Bias (Source: Beinghuman.org)

- Give more weight to information that confirms what we already believe
  - Automatic unconscious way our brains process information
  - Selectively remember information that confirms what we already think
  - When we approach new information, we interpret it in a biased way
  - Spin news story so it vindicates their own beliefs?
- We subconsciously only pay attention to the information that confirms what is already known

You would think this would help ensure viable estimates but… Its what we believe, not necessarily what is reality

Negativity Bias (Being Human.org)

- Unconsciously pay give more weight to negative experiences than positive ones
- Brains react powerfully to negative information than they do to positive information
- Daniel Kahneman explained:
- “The brains of humans and other animals contain a mechanism that is designed to give priority to bad news. By shaving a few hundredths of a second from the time needed to detect a predator, this circuit improves the animal’s odds of living”
- More important for our ancestors to be able to avoid a threat quickly than to gain a reward

Again, this should yield viable estimates but is usually overridden

Loss Aversion Bias (Source BeingHuman.org)

- Tendency to strongly prefer avoiding a loss to receiving a gain
  - Explains making same irrational decisions over and over
- Kahneman: Experiment giving one third of the participants mugs, one third chocolates, and one third neither
  - Option of trading
    - 86 percent who started with mugs chose mugs
    - 10% who started with chocolate chose mugs
    - 50% who started with nothing chose mugs

- Throwing good money after bad (sunk cost fallacy) i[s a] perfect example of loss aversion
- To avoid feeling the loss we stick with our plan, hoping for a gain, even when that just leads to a bigger loss

Explains why it is so hard to kill a failing program

Affect Heuristic Bias (Source: Beinghuman.org)

- Involuntary response to a stimulus that speeds up the time it takes to process information
  - If we have pleasant feelings, we see benefits high and risks low, and vice versa
  - affect heuristic behaves as a first and fast response mechanism in decision-making
  - Helpful in life or death situations where time was of the absolute essence.
- **System 2** The analytic, rational system of the brain is relatively slow and requires effort
- **System 1** The experiential system is different—speedy, relying on emotional images and narratives that help us to estimate risk and benefit.

Hopefully estimates elicit system 2… But often are off the cuff via system 1

Thinking Fast & Thinking Slow (Source: Kahneman)

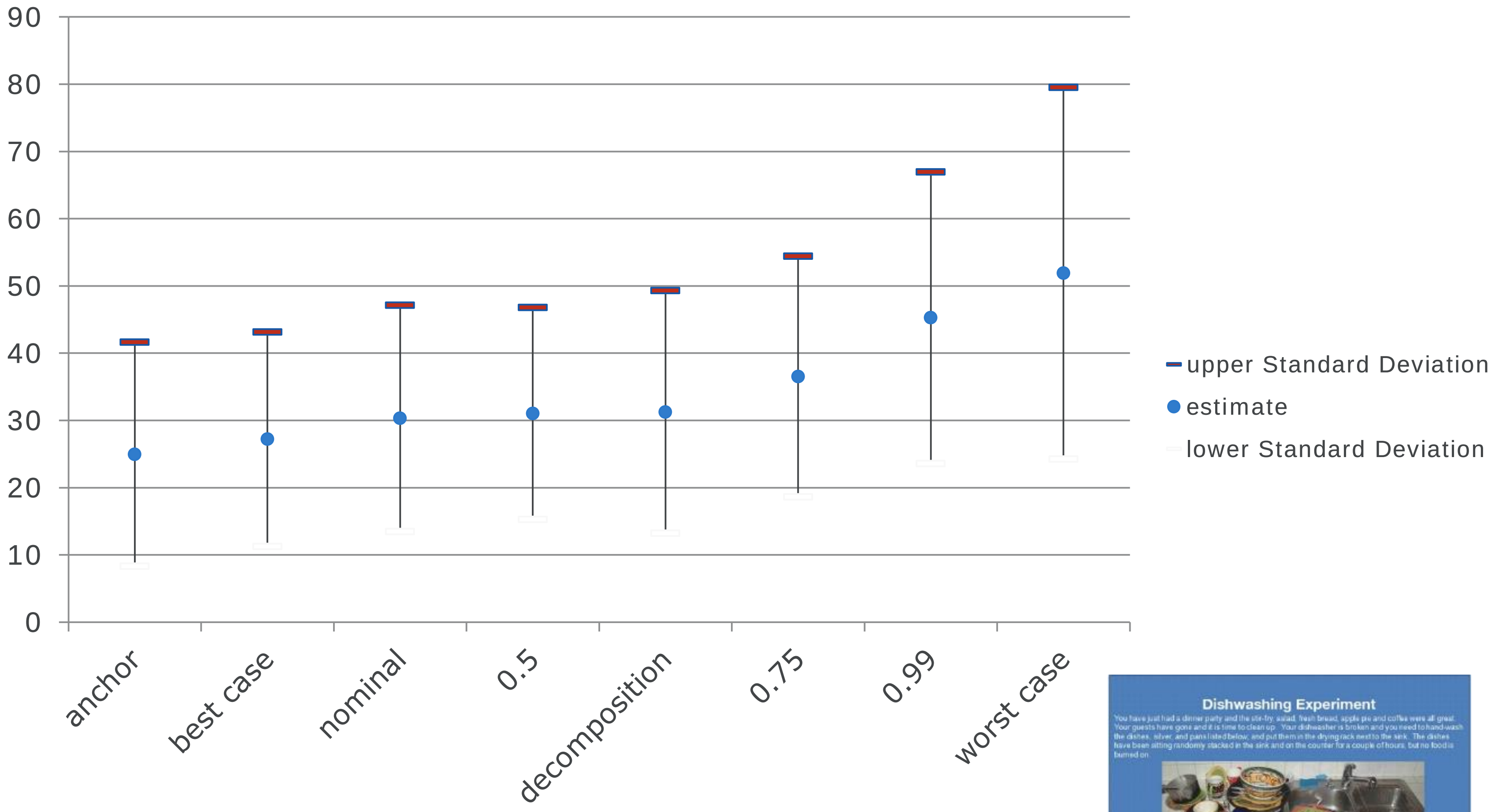
System 1: Thinking Fast	System 2: Thinking Slow
• Operates Automatically • No effort • Quick • No voluntary control	• Allocates attention to mental activities that demand it • Complex computations
• Coherent interpretation of what is going on	• Good at balancing probabilities but often indecisive
• Intuitive answers quickly	• Takes over when System 1 can't process the data • If the person is willing • Can correct or override System 1 if it determines intuition is wrong

Illusion of Control (Source: BeingHuman.org)

- Tendency to overestimate their influence over outcomes that they cannot affect
- Psychologist Ellen Langer Subjects given lottery tickets; either at random or allowed to choose their own
  - Had chance to trade tickets for others that had a higher chance of paying out.
  - Subjects who chose ticket were less likely to part with it than those who had a random ticket
  - Subjects felt their choice of ticket had some bearing on the outcome—demonstrating the illusion of control.
- Illusion of control especially strong in stressful and competitive situations, like gambling or financial trading or ESTIMATING

Illusion of control can lead bad decisions or irrational risks

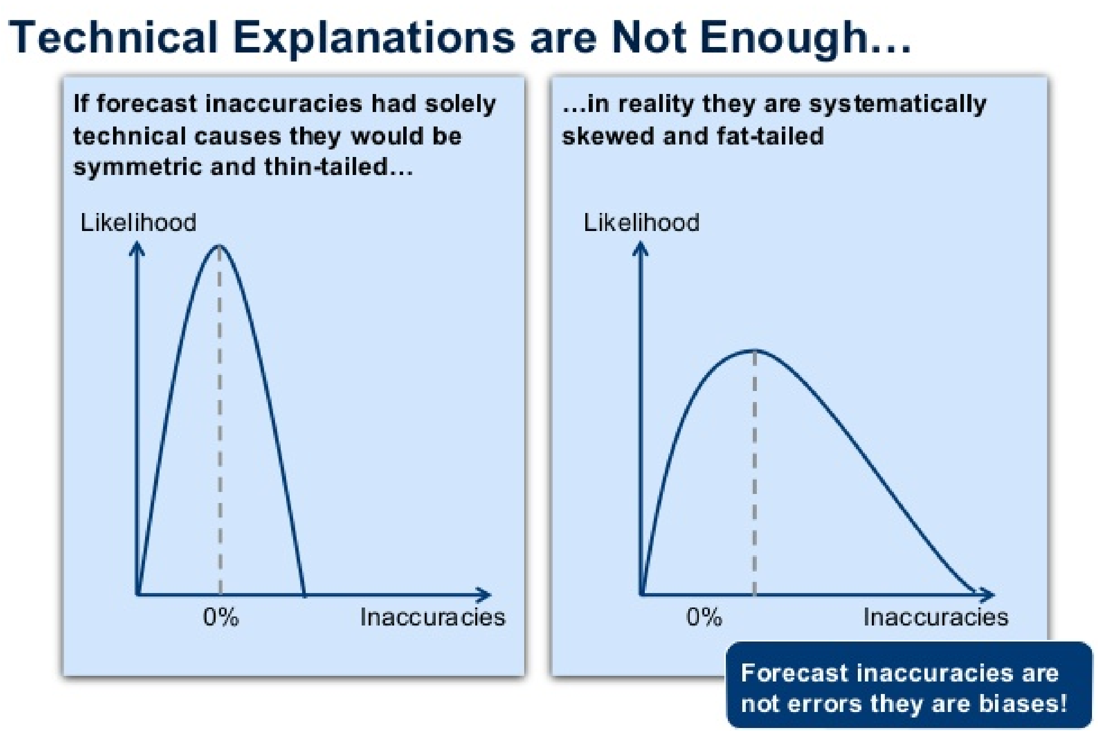
Dishwashing Estimation Bias Study Summary (Source: JPL http://www.slideshare.net/NASAPMC/arthurchmielewski)



Explanations for Poor Estimating

(Adapted From Source Master Class on Risk, Flybjerg, 2013)

1. Technical: Inadequate data & Models (Vanston)
2. Psychological: Planning Fallacy, Optimism Bias - causes belief that they are less at risks of negative events
3. Political / Economic: Strategic misrepresentation - tendency to underestimate even when experienced with similar tasks overrunning (Flyvberg)



Draw Out Range By Obtaining 3 Estimates

- Optimistic value (s_{opt})
- Most likely value (s_m)
- Pessimistic value (s_{pess})

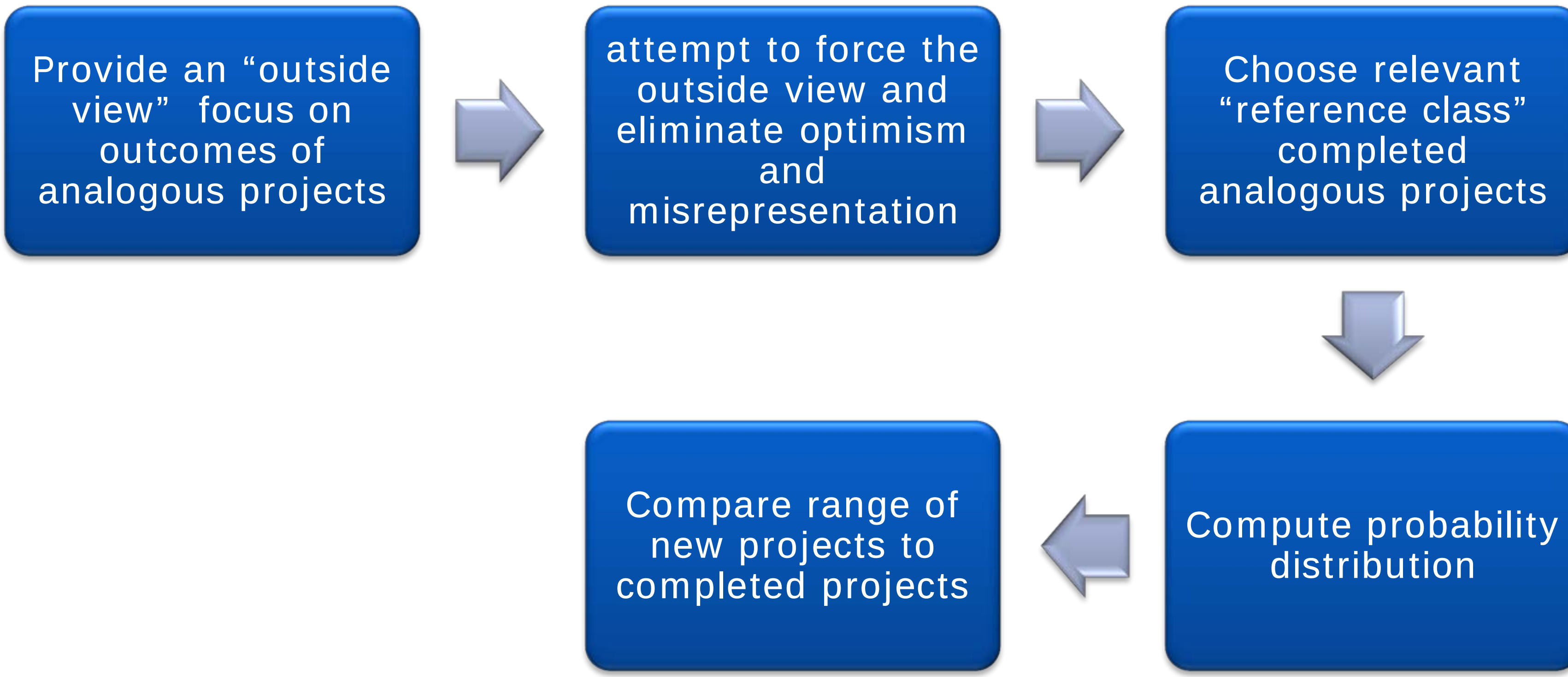
- Expected value (*EV*)

$$EV = \frac{(s_{opt} + 4s_m + s_{pess})}{6}$$

5 Levels of Risk Management
(Adapted from Flyvbierg)

1 Opinions	2 Benchmarking	3 Diligence	4 Rigorous Estimating	5 Risk Analysis
As unbiased as possible	Comparing to viable database	Estimate review	Parametric Relative Reference Class Forecasting	Risk management Black Swan mitigation

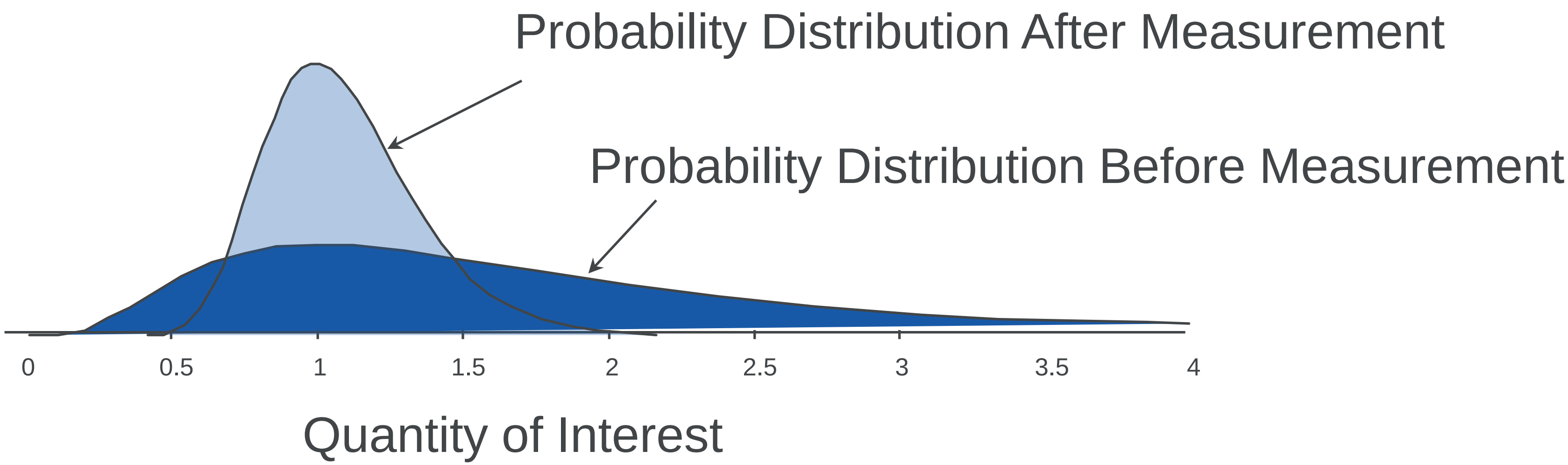
Reference Class Forecasting (adapted from http://www.slideshare.net/assocpm/a-masterclass-in-risk)



Best predictor of performance is actual performance of implemented comparable projects (Nobel Prize Economics 2002

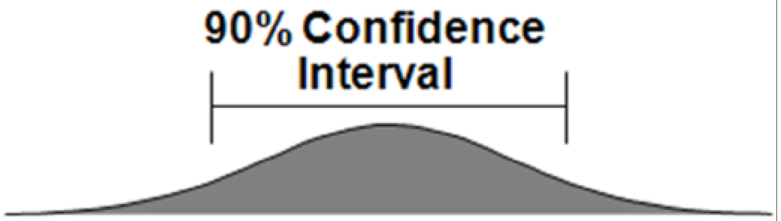
Hubbard: Measure To Reduce Uncertainty

- Perception that measurement is a point value is a key reason why many things are perceived as “immeasurable”
- Measurement: Quantitatively expressed reduction in uncertainty based on observation



Assumptions, Change Drivers & Expert Judgment Need Caution (Source: Hubbard)

- Most people are significantly ***overconfident*** about their estimates ... especially educated professionals



Group	Subject	% Correct (target 90%)
Harvard MBAs	General Trivia	40%
Chemical Co. Employees	General Industry	50%
Chemical Co. Employees	Company-Specific	48%
Computer Co. Managers	General Business	17%
Computer Co. Managers	Company-Specific	36%
AIE Seminar (before training)	General Trivia & IT	35%-50%
AIE Seminar (after training)	General Trivia & IT	~90%

(AIE = Hubbard Generic Calibration Training)

Used with permission from Douglas Hubbard

Example - Pairwise Comparisons

- • Consider following criteria

Purchase Cost | Maintenance Cost | Gas Mileage

• Want to find weights on these criteria

• AHP compares everything two at a time

(1) Compare Purchase Cost **to** Maintenance Cost

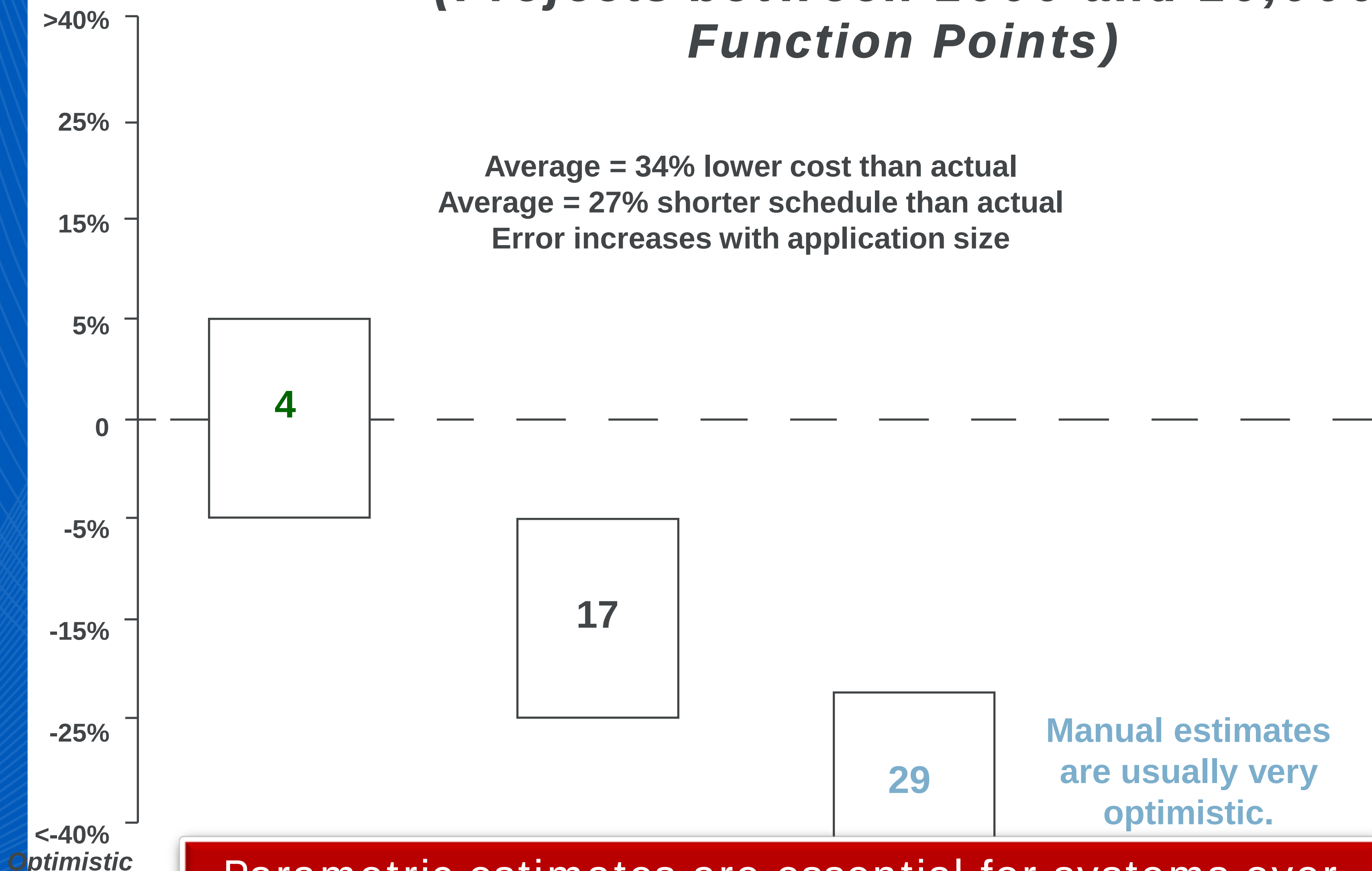
– Which is more important?

Say purchase cost

– By how much? Say moderately → 3

ACCURACY RANGES FOR 50 MANUAL ESTIMATES (Source: Capers Jones)

- ***(Projects between 1000 and 10,000 Function Points)***



Parametric estimates are essential for systems over 1,000 function points