Preview of ICEAA SCEBoK 1.0: Software Cost Estimating Body of Knowledge

Carol Dekkers, Lead Author
Quality Plus Technologies, inc.

ICEAA International Cost Estimating and Analysis Association

SCEBoK

WHO AM I?

Carol Dekkers, PMP, CFPS (Fellow), P.Eng.

Lead author of ICEAA SCEBoK
IFPUG Past President
ISO project editor
Founder, Quality Plus Technologies, Inc.

Consultant. Author. Speaker. Instructor

ICEAA Software Cost Estimation Body of Knowledge: SCEBoK Outline

LESSON 0: INTRODUCTION TO CURRICULUM

LESSON 1: IMPORTANCE AND MOTIVATION FOR SCEBOK

LESSON 2: SOFTWARE DEVELOPMENT PARADIGMS

LESSON 3: SCEBOK FIVE-STEP ESTIMATING PROCESS

LESSON 4: ESTIMATING CUSTOM SOFTWARE DEVELOPMENT

LESSON 5: SOFTWARE SUSTAINMENT

LESSON 6: ESTIMATING PROCURED SOFTWARE SOLUTIONS

LESSON X: SOFTWARE SIZE

LESSON Y: PRODUCTIVITY

LESSON Z: COMMERCIAL ESTIMATING MODELS

SCEBoK

LESSON 0:
INTRODUCTION TO
CURRICULUM

Preview of ICEAA SCEBoK 1.0

- **Special thank you to the SCEBoK Senior Advisors and Leadership Team**
  - Kevin Cincotta, ICEAA Certification and MITRE
  - Rick Collins, ICEAA President and Technomics
  - Dave Brown, Technomics
  - Wilson Rosa, DHS CAD
  - Christian Smart, ICEAA Board and Galorath
  - Megan Jones, ICEAA Executive Director
  - Eric van der Vliet, Nesma and CGI
  - Harold von Heeringen, Nesma and Metri

- **ICEAA SCEBoK Review Group (ISRG)**

SCEBoK

LESSON 0:
INTRODUCTION TO
CURRICULUM

SCEBoK's Global Audience

(SCEBoK Terms of Reference, May 2020)

SCEBoK audiences consist of cost estimators and/or software engineers from various industries, including (and not limited to):

- Original Equipment Manufacturers (OEMs)
- Prime contractors & subcontractors
  - E.g., government(s), defense, intelligence and civil agency projects
- Commercial organizations & IT departments
  - E.g., banks, insurance companies, etc.
- Consulting firms
  - E.g., consultants to OEMs, commercial or government organizations
- Government and quasi-government organizations (e.g., Federally Funded R&D Centers)
- Academic institutions

SCEBoK

LESSON 0:
INTRODUCTION TO
CURRICULUM

SCEBoK Inclusions/Exclusions

(SCEBoK Terms of Reference, May 2020)

SCEBoK does:

- Provide the user, whose background and expertise are intended to be based in cost estimating and analysis, with an understanding of software estimating that will complement and enhance their cost estimates and analyses.

SCEBoK does NOT:

- Endorse any particular method of software sizing as superior or inferior to another;
- Endorse any particular software development methodology, software estimating methodology or vendor;
- Prescribe the essential considerations in software cost estimating – however, it DOES provide guidance.

SCEBoK

LESSON 0:
INTRODUCTION TO
CURRICULUM

Pre-requisite Knowledge: ICEAA CEBoK Modules

Contents [hide]

1 Unit I - Cost Estimating
- 1.1 Module 1 - Cost Estimating Basics
- 1.2 Module 2 - Costing Techniques
- 1.3 Module 3 - Parametric Estimating

2 Unit II - Cost Analysis Techniques
- 2.1 Module 4 - Data Collection and Normalization
- 2.2 Module 5 - Inflation and Index Numbers

3 Unit III - Analytical Methods
- 3.1 Module 6 - Basic Data Analysis Principles
- 3.2 Module 7 - Learning Curve Analysis
- 3.3 Module 8 - Regression Analysis
- 3.4 Module 9 - Cost and Schedule Risk Analysis
- 3.5 Module 10 - Probability and Statistics

4 Unit IV - Specialized Costing
- 4.1 Module 11 - Manufacturing Cost Estimating
- 4.2 Module 12 - Software Cost Estimating

5 Unit V - Management Applications
- 5.1 Module 13 - Economic Analysis
- 5.2 Module 14 - Contract Pricing
- 5.3 Module 15 - Earned Value Management
- 5.4 Module 16 - Cost Management

- **Basic knowledge of cost estimating content as highlighted**
- **Available to ICEAA members https://wikidev.iceaaonline.com/wiki/Main_Page**

SCEBoK

Why ICEAA SCEBoK ?

Software presents a number of unique challenges for the estimator to get a Realistic / Close estimate

- Understanding software cost estimation is critical because software is increasingly part of almost every program estimate
- Paradigms, software growth, packaged solutions, cost drivers, and correct usage of historical data are pre-requisites to realistic estimates
- Status quo is not good enough → need formal estimating process, historical data, and repeatable practice

LESSON 1: IMPORTANCE AND MOTIVATION FOR SCEBOK

Lesson 1: Importance and Motivation for Software Cost Estimating

Lesson outline:

1. The importance of software cost and schedule estimation
2. The impact of software estimation on project outcomes
3. Setting the stage for the remainder of the SCEBoK lessons:
   - Develop an understanding of what are (and how to create) credible and defensible software estimates
   - Define common terminology useful to a software cost estimator
   - Introduce the software life cycle and the fit with software development and acquisition decision making (whether contracted or internal)

SCEBoK

LESSON 2: SOFTWARE DEVELOPMENT PARADIGMS

Lesson 2: Software Development Paradigms

Lesson outline:

- **Terminology and definitions of what is software development**
- **Introduction to software development paradigms:**
  - Comparison and contrast of paradigms in use today
- **Cost considerations of software development paradigms:**
  - What are major considerations by paradigm and how to differentiate the software cost estimating implications of each software development paradigm

SCEBoK

LESSON 2: SOFTWARE DEVELOPMENT PARADIGMS

Adaptive versus Predictive paradigms: overview

Differentiator	Agile methods	Predictive methods
Focus of work	Change-driven based on product value: fixed time and cost, estimated scope	Project plan: fixed scope, estimated cost and duration
Frequency of deployment	Iterative, early & frequent (2-3 weeks)	One big release (end of project)
Quality	Continuous inspection and testing during every iteration (impact on rework)	Testing phase inspects out poor quality at the end (impact on rework)
Customer involvement	Co-located, daily review via product owner	Interaction with customers at beginning and end
Risk of changing requirements	Minimizes risk by engaging customer to prioritize requirements to be developed first	Fixed requirements with high risk due to incomplete or incorrect requirements or change
Development teams	Self-organized, may split effort between Development and Operations	Hierarchical or matrix, fully dedicated to development
Operations and Security Considerations	With DevSecOps, continuous deployment, continuous testing during development. Can reduce integration costs	Little to no consideration of operations or security during development. Separate teams. Operations and security concerns handled post-development
Prototypical contract type(s)	Time and materials (T&M) or cost-reimbursable	Firm fixed price (FFP) or fixed price with incentives

SCEBoK

Adaptive versus Predictive paradigms: cost considerations[1]

Topic	Agile	Predictive/ waterfall	Hybrid-agile (Partly predictive/partly agile)
Fixed	Cost & schedule	Scope (requirements)	Initial high-level scope fixed, but is flexible to prioritization and change during development
Estimated	Scope (features)	Cost & schedule	Cost & schedule initially estimated, but becomes fixed during development
Driver	Change-driven	Plan-driven	Hybrid
Development risks[1]	Cost & schedule mostly fixed. Delivered size may fall short	Scope fixed → cost & schedule overruns	Cost & schedule mostly fixed during build. Delivered scope may fall short

Adapted from https://www.process.st/waterfall-vs-agile/

LESSON 3: SCEBOK FIVE-STEP ESTIMATING PROCESS

Lesson 3: SCEBoK Five-Step Estimating Process

Lesson outline:

- **Identify and discuss types of estimates for custom software development efforts**
- **Present the SCEBoK 5-step software estimating process:**
  1. Develop the scope of the estimate;
  2. Collect and analyze historical data;
  3. Create the software estimate;
  4. Adjust the estimate for risk and uncertainty; and
  5. Document and present the estimate

SCEBoK

LESSON 4: ESTIMATING CUSTOM SOFTWARE DEVELOPMENT

Lesson 4: Estimating Custom Software Development

Lesson outline:

0. **High-level review:**
   - Software Size and Productivity
   - SCEBoK Five-Step Estimating Process
1. **Three ways to create the software estimate using Estimating Techniques:**
2. **Overview of estimating approaches**
3. **Creating a software development estimate based on SCEBoK Five-step Estimating Process and four estimating approaches**
4. **Schedule estimate**
5. **Time-phasing the estimate**
6. **Cross-checking the estimate**
7. **Outside of the Five-step Estimating Process: Reviewing Estimates prepared by others**

SCEBoK

LESSON 4: ESTIMATING CUSTOM SOFTWARE DEVELOPMENT

Three Ways to Create a Software (Development) Estimate

Way	Process	Associated approach	Pre-requisites for Selecting
1st	• Estimate size • Estimate effort based on size (with productivity implied) • Estimate schedule and cost based on effort	1. Parametric (Derive CER) 2. Parametric (Published CER)	• Historical data • Software size estimate available • Known criteria
2nd	• Estimate size, productivity • Estimate effort based on size and productivity • Estimate schedule and cost based on effort	3. Analogy (Productivity-based)	• Analogy available • Software size estimate available • Known criteria
3rd	• Estimate effort, schedule and cost based on expert opinion	4. Wideband Delphi Expert Opinion	• Lack of historical data • May not know software size
Cross-check[1]	• Use additional approaches (above) or published Rules-of-Thumb • Used to support primary approach	5. Commercial Estimating Models 6. Rules of thumb	• Known criteria (see Lesson Z) • Rough order of magnitude

1. Cross-check is not another approach, but rather is a means to validate the realism of an estimate created in one of other ways above. Use a second approach, a commercial software estimating model (Lesson Z) or published Rules-of-Thumb to validate estimate realism

SCEBoK

Lesson 5: Software Sustainment

LESSON 5:
SOFTWARE
SUSTAINMENT

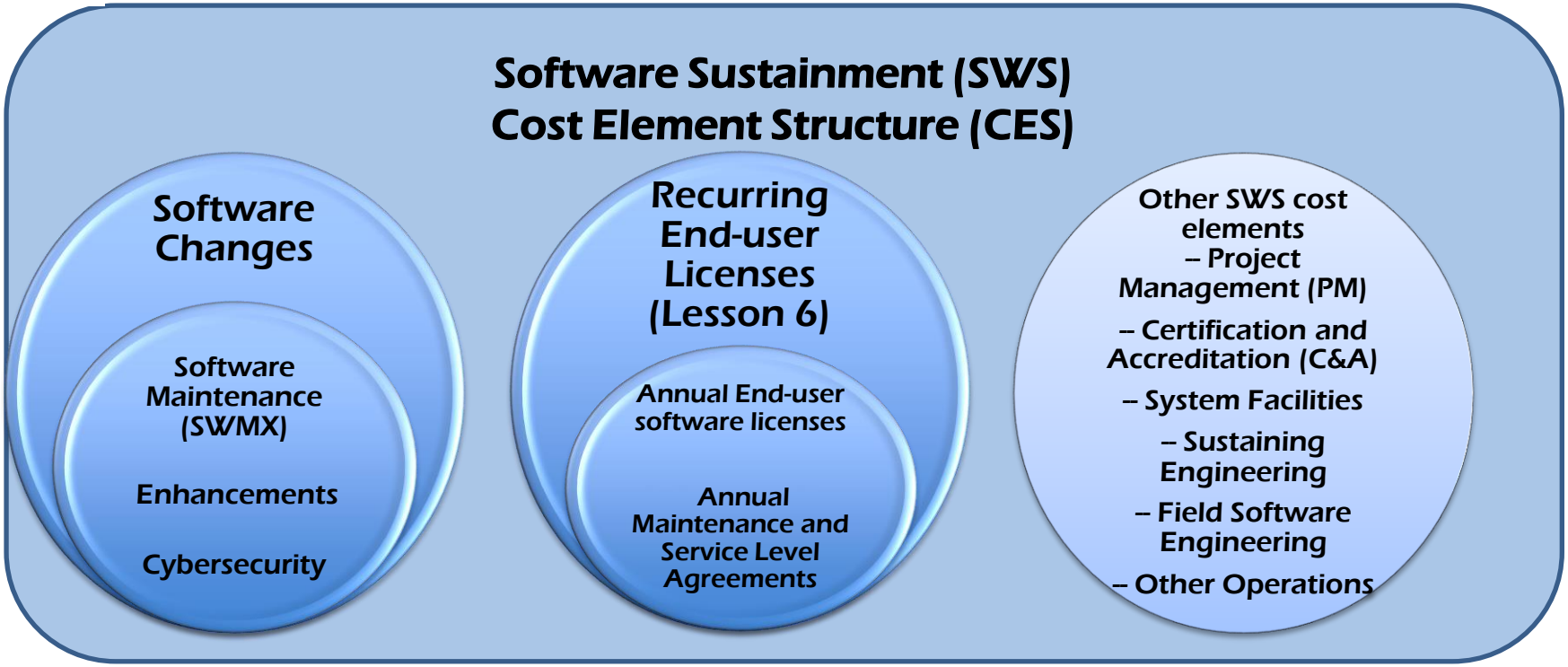
Lesson outline:

1. Definitions and Terminology: Software Sustainment vs Software Maintenance
   a. Software Sustainment (SWS)
   b. SWS Cost Element Structure (CES)
   c. Software Changes: Software Maintenance (SWMX), Enhancements, and Cyber Security
2. The Importance of Software Sustainment
   a. Data Normalization
   b. Drawing the Line Between Development (Investment) and Sustainment
   c. Reasons for the High Costs of Software Sustainment
3. Cost Estimating Techniques
4. SWS Risk/Uncertainty Considerations
5. Implications of Dev(Sec)Ops on Structure and Time-Phasing of Estimate
6. A Note About Obsolescence
7. Quick Reference/Rules of Thumb
8. Lesson Summary

SCEBoK

LESSON 5: SOFTWARE SUSTAINMENT

Software Sustainment (SWS) Cost Element Structure (CES)[1]



Adapted from the SWS WBS v5.0 from the New Army Software Sustainment Cost Estimating Results DASA-CE by Cheryl Jones, James Doswell, et al, presented at ICEAA May 2019

SCEBoK

LESSON 5:
SOFTWARE
SUSTAINMENT

Cost Estimating Methods for Sustainment and Maintenance

	Cost Method[2]	Definition	Associated SWS CES Element(s)
1.	Cost Factor-based	Approaches that estimate software sustainment cost as a percentage, or factor, of some other (estimated) cost.	Overall SWS
2.	Size-based Productivity, and Annual Change Traffic (ACT)-based	Approaches that estimate software product changes costs based on delivered software size and productivity or on the annual amount of code touched or changed	Typically Software Changes
3.	Parametric CER	Approaches that allow the software cost estimator to create their own CER based on Total Software Changes (TSC) or other measures	Typically Software Changes
4.	Commercial Estimating Models: (See Lesson Z)	Commercial Estimating Models give estimated hours associated with each type of SWMX based on the inputs described previously[1] (e.g., size, complexity, capability)	SWS (Life Cycle Costs) or SWMX
5.	Past Funding-based	Approaches that rely on FTE historical budget data to estimate software sustainment as steady state level-of-effort (LOE) task	Overall SWS

1. In Lesson 4: Creating the Software Estimate for Software Development
2. Note that a more-detailed estimating method would take multiple values into account, such as actual delivered size, complexity, etc.

SCEBoK

LESSON 6: ESTIMATING PROCURED SOFTWARE SOLUTIONS

Lesson 6: Procured Software Solutions

Lesson outline:

A. Review: Drawing the Lines between Investment and Sustainment:
   - SCEBoK Software Delivery Continuum, SCEBoK CES

B. Introduction to procured software solutions:
   - What is COTS[1] software?

C. Identify major types of procured software solutions, and cost drivers and estimating approach(es) for each:
   1. COTS Purchase
   2. Software as a Service (SaaS))
   3. Service-Oriented Architecture (SOA)
   4. Enterprise Resource Planning (ERP)
   5. Enterprise Data Warehouse (EDW), Data Mart, Operational Data Store (ODS)
   6. Hybrid solution

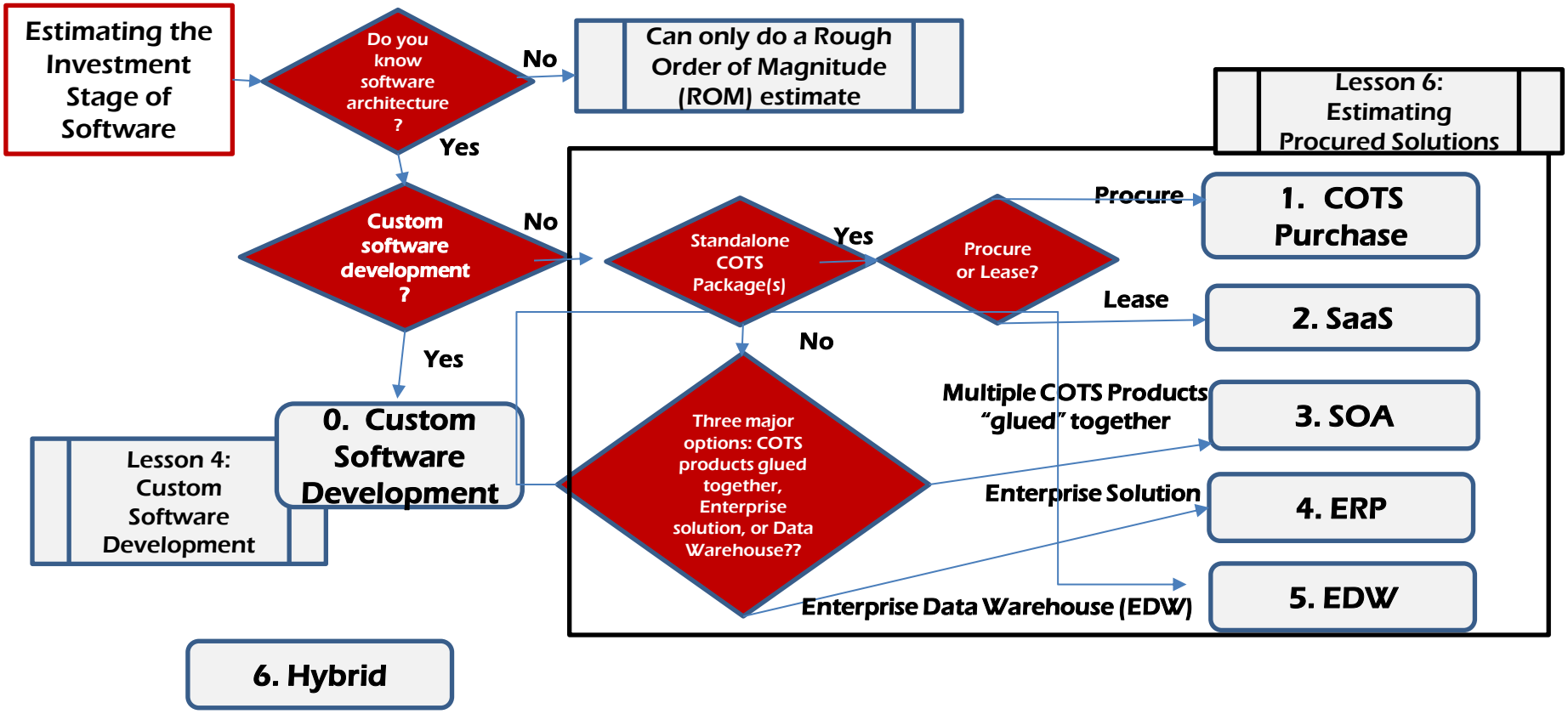
D. Consideration of Cost Growth in Procured Software Solutions

E. Software sustainment considerations specific to procured software solutions

F. Lesson Summary

SCEBoK

Identify the Type of Procured Software Solution



LESSON 6: ESTIMATING PROCURED SOFTWARE SOLUTIONS

Estimating approaches for Procured Software Solutions

#	Solution	Known as	Lesson	SCEBoK estimates the:	Cost Drivers	Estimating approaches
0	Custom software: design & build the solution	**Custom software development**	4	• Effort for software development (software requirements analysis, design, build, test, install software) → cost • Schedule (duration)	• Software size (FP, ESLOC, other) • S/W Complexity • Dev team Capability	• Parametric CER (Custom or Published) • Analogy • Expert Opinion
1	Single COTS	**COTS purchase**	6	• Procurement cost	• Vendor quotes (# of users, sites)	• Build-up of quotes
2	Lease the functionality	**Software as a Service (SaaS)**	6	• Leasing cost by duration, or number of users or sites	• Vendor quotes (# of users, months)	• Vendor quotes
3	Multiple COTS	**Service-Oriented architecture (SOA)**	6	• Effort to develop glue code→ cost • Schedule (duration)	• # of interfaces that need to be built	• CER based on interfaces
4	Enterprise Solution (Oracle, SAP)	**Enterprise Resource Planning (ERP)**	6	• Effort for integration (configure /tailor /customize RICE(FW) objects → cost • Schedule (duration)	• RICE(FW) objects	• CER based on RICE-FW objects
5	Executive Info system	**Enterprise Data Warehouse (EDW), Data Mart, ODS**	6	• Effort for Extract Transform Load routines (ETL) from source systems	• # of migrated tables	• CER based on migrated tables
6	Hybrids	**Hybrid solution**	4/6	• Use combination of approaches	• See above	• Combination

1. RICE = Reports, Interfaces, Conversions, Extensions, (Forms and Workflows) objects in ERP software

SCEBoK

LESSON X:
SOFTWARE SIZE

Lesson X: Software Size

Lesson Outline:

- Software size as a cost driver for software development
- Dimensions of software size, methods, and units of measure:
  1. Physical size
  2. Functional size and Extensions to functional size for non-functional software requirements
  3. Relative effort measures (agile software development)
  4. RICE(FW)[1] objects (for Enterprise Resource Planning procured software solutions)
- Software size considerations
  - Choosing a Sizing Approach
  - Conversions between size units and Rules of Thumb
  - Questions to Ask
- Software sizing tools:
  - Commercially and freely available Source Lines of Code (SLOC) counters: (University of Southern California Unified Code Counter (UCC))
  - Function Point estimating tools: CADENCE, ScopeMaster

SCEBoK

LESSON X:
SOFTWARE SIZE

Software Size Dimensions and Units of Measure

1. **Physical size: Source Lines of Code (SLOC)**
2. **Functional Size**
    A. International Function Point Users Group (IFPUG) Function Points (ISO/IEC 20926) **
    B. Nesma Function Points (ISO/IEC Standard ISO/IEC 24750) **
    C. Common Software Measurement International Consortium (COSMIC) Function Points (ISO/IEC Standard) **
    D. Simple Function points
    E. Object Points
    F. Use Case Points
    G. Requirements (Software Engineering Institute)

    Non-functional extensions to functional size

    H. IFPUG Value Adjustment Factor (VAF)
    I. IFPUG Software Non-functional Assessment Process (SNAP) points
    J. Use Case Points Technical Factor (TF) and Environmental Factor (EF)
3. **Relative effort: (Agile development)**

    Three methods for estimating Story Points:

    A. Planning poker
    B. T-shirt sizing
    C. Time buckets
4. **RICE(FW) – Reports, Interfaces, Conversions, Extensions, (Forms, Workflows) – for Procured software solutions**

** Conformant with ISO/IEC 14143-1 Functional Size Measurement Part 1: Definition of concepts

SCEBoK

Lesson Y: Productivity

Lesson Outline:

1. Productivity as a concept
2. Define Productivity in the context of software development cost estimating
3. Ways of measuring productivity; specify your measurement (unless otherwise specified)
4. Productivity as an input vs. output (can cross-check)
5. Relationship between size and productivity

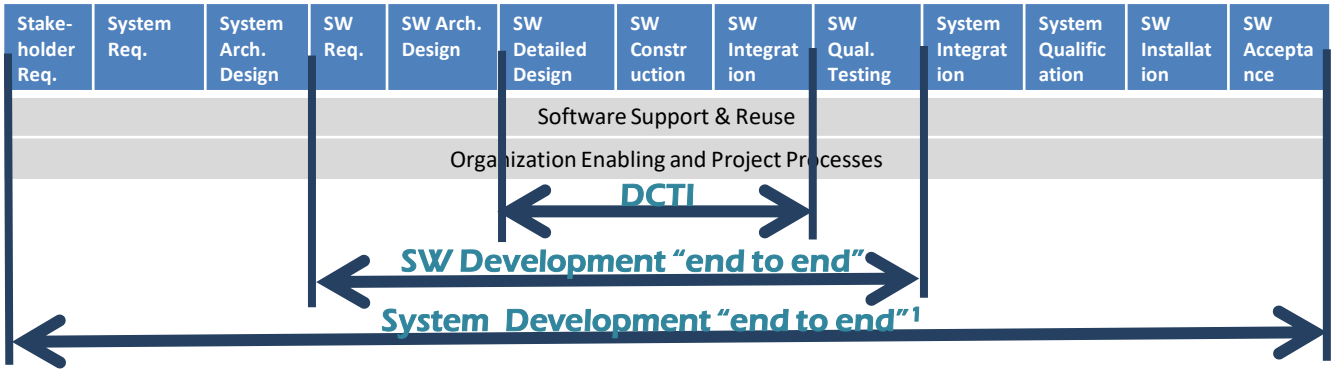
SCEBoK

LESSON Y:
PRODUCTIVITY

Ways of Measuring Productivity: Considerations

The graphic below illustrates how productivity factors can be defined over differing portions of the overall development process

- Design, Code, Test, Integration (DCTI) factors cover the "core" parts of the process; other activities must be estimated separately or significant omissions will occur in the estimate
- Software "end to end" productivities attempt to cover all "software-specific" activities; higher level systems engineering activities must be estimated separately where relevant



Work content covered by the productivity factor must be well-defined and well-understood!

SEBoK

LESSON Y:
PRODUCTIVITY

Ways of Measuring Productivity

	Definition (of Productivity)	Typical Estimating Technique	Source
1	1/ (A* Size$^{(E-1)}$) in Custom CER where effort = A*Size^{E}	Custom Cost Estimating Relationship (CER)	Defense Acquisition University (DAU)
2	A general term for all of the non-size cost drivers (in COCOMO II)	Generalized CER such as COCOMO II	COCOMO II manual
3	A general term for all of the non-size cost drivers (general)	Parametric	Defined by parametric model developers (e.g., commercial estimating model vendors)
4	Interchangeably with COCOMO II Effort Multipliers (EM)	COCOMO II equation	Non-specific (as in "productivity and cost drivers"[1])
5	"A" in COCOMO II equation	COCOMO II equation	COCOMO II manual (as "productivity constant").
6	Size / effort	Analogy	Stutzke, Jones, etc., intuitive output / input
7	Effort / size	Analogy	ISBSG Project Delivery Rate (PDR); Any source citing Effort = Size * Productivity

1. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.366.221&rep=rep1&type=pdf

SCEBoK

More in-depth Productivity Concepts (Creative Commons): Rethinking Productivity in Software Engineering (2019)

LESSON Z: COMMERCIAL ESTIMATING MODELS

Lesson Z: Commercial Estimating Models

Lesson Outline:

1. **Introduction to Commercial Estimating Models**
2. **Overview of four models:**
   1. COCOMO II Web Tool
   2. Galorath Inc: SEER-SEM ®
   3. PRICE Systems: TruePlanning for Software®
   4. Quantitative Software Management (QSM): SLIM Estimate®

SCEBoK

SCEBoK: Using Data to Create Realistic Estimates (and Better Results)

Cost estimating best practices *CAN BE TAILORED* to software:

- **Estimating Maturity Model (formal methods)**
- **Cone of uncertainty**
- **Foundation of solid historical data**
- **Data analysis and normalization**
- **Planning for cost and schedule growth**
- **Cross checks, sensitivity analysis, risk and uncertainty**

SCEBoK

A Final Note…

'The software industry has the worst metrics and measurement practices of any industry in human history' – Capers Jones (2018)[1]

"SCEBoK will help create realistic data-based estimates. Over time, this leads to more, successful projects, and (hopefully) better metrics." – Carol Dekkers, March 2021

1. Source: Capers Jones, ***Quantifying Software – Global and Industry Perspectives***, 2018

SCEBoK

How can YOU support SCEBoK?

- Join ICEAA, promote SCEBoK
- Participate in ICEAA training
- Volunteer:
  - Write certification questions
  - Provide content/review materials
  - Teach a lesson at ICEAA

Contact: Kevin Cincotta, Carol Dekkers, Megan Jones, ICEAA Board members

SCEBoK

Carol Dekkers

813-816-1329

caroldekkers@gmail.com

www.qualityplustech.com

SCEBoK:

www.iceaaonline.com

SCEBoK

SCEBoK

LESSON 0:
INTRODUCTION TO
CURRICULUM

SCEBoK Objectives (3 of 3)

(SCEBoK Terms of Reference, May 2020)

5. SCEBoK will be a how-to for reviewing another party's
   - Estimate of software development effort, cost and schedule
   - Estimate of software maintenance cost and schedule

   Note: The ability to prepare an estimate (i.e., point 4) is necessary, but not sufficient to critically review an estimate; therefore, content specific to reviewing is required

6. SCEBoK will be commercial and non-commercial software cost and schedule estimating tool agnostic

7. SCEBoK will be software sizing technique agnostic

SCEBoK

LESSON 1: IMPORTANCE AND MOTIVATION FOR SCEBOK

Status Quo: Standish Group CHAOS reports (since 1996)

Standish Group CHAOS Report defines project outcomes:

SUCCESS: On-time, on-budget, with all features

CHALLENGED: Delivered, but... late, and/or over-budget, and/or missing features

FAILED: Not delivered or cancelled

SCEBoK

Software Development Paradigms Pertain to Software Development

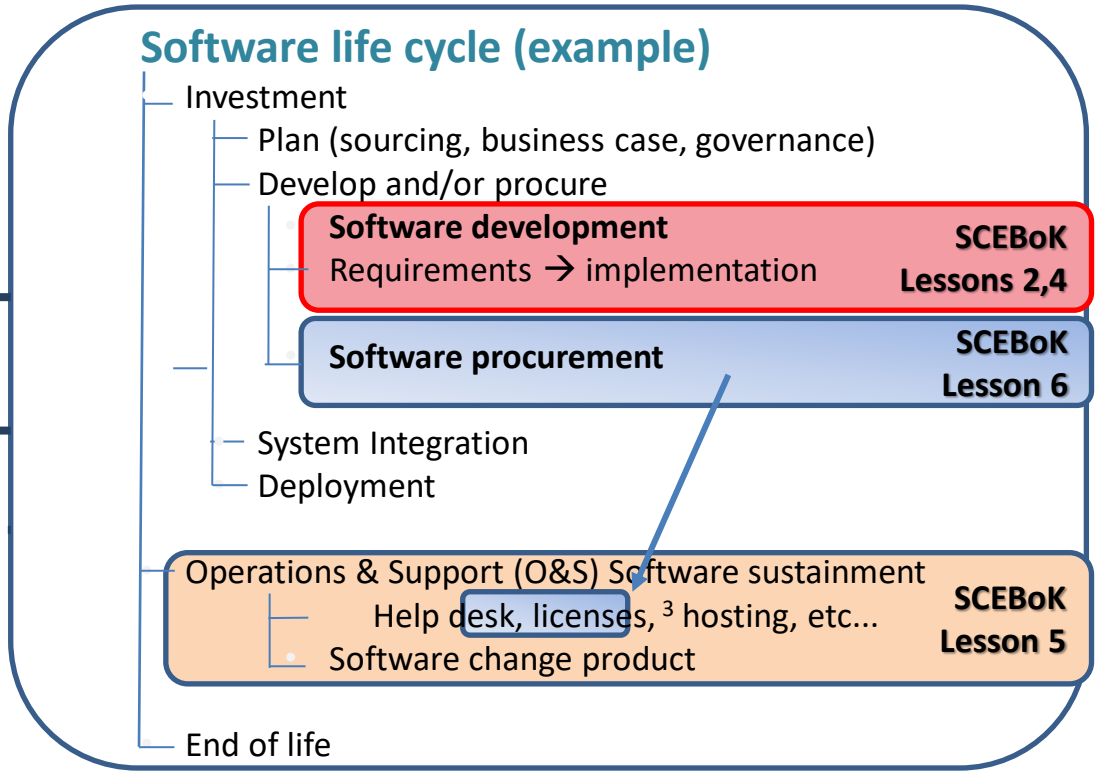
Software-intensive program[1]

Investment:

- Program/project management
- Systems engineering
- BPR/ Change management
- System Development
- System Procurement
  - Hardware (make and/or buy)
  - Software (buy)
- System level integration & test
- System deployment/implementation

Operations & support (O&S)[2]

- Help desk/service desk support
- Technology refresh/upgrade
- System maintenance



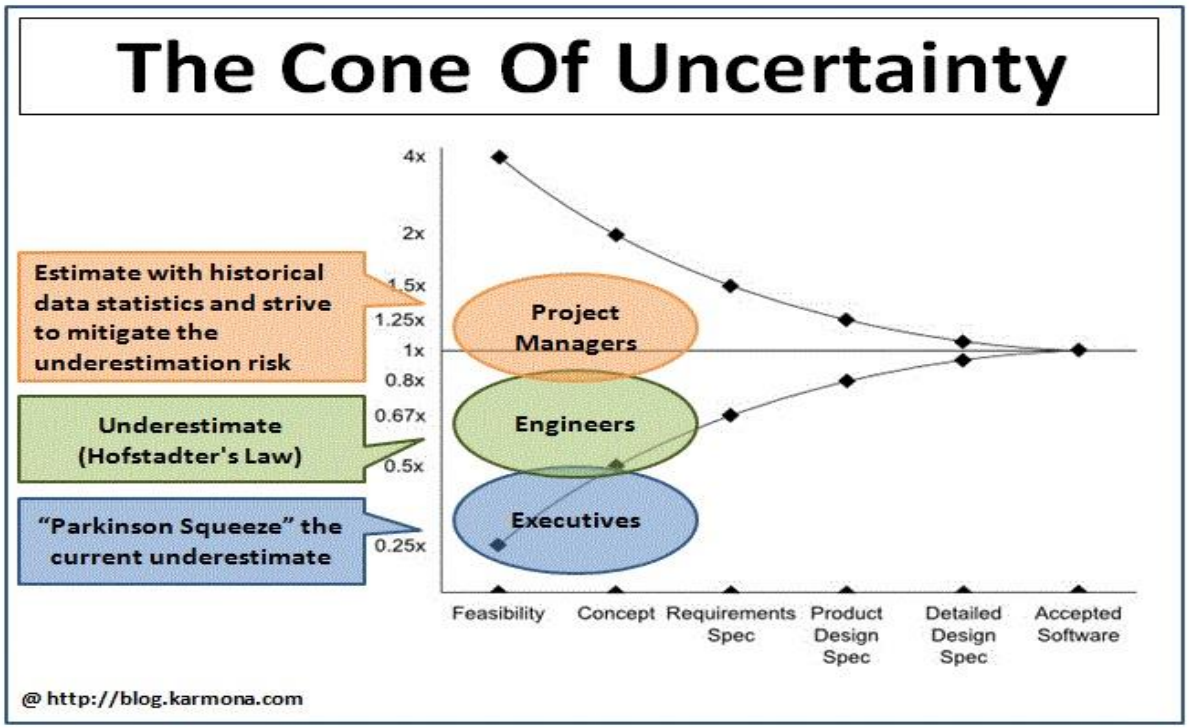
1. Standard IT LCC WBS V5 from US Department of Homeland Security
2. O&S contains continuation of many investment categories + those new ones listed here... See definitions in notes
3. Software obsolescence is of growing concern in O&S, see lesson 6 for research and further information.

SCEBoK

Software-specific Cone of Uncertainty

What is Uncertain?

- Software Size
- Software Complexity
- Team capability
- Schedule constraints
- Team size
- Productivity
- Relationships between:
  - Size and effort
  - Size and productivity
- Historical data (quality)



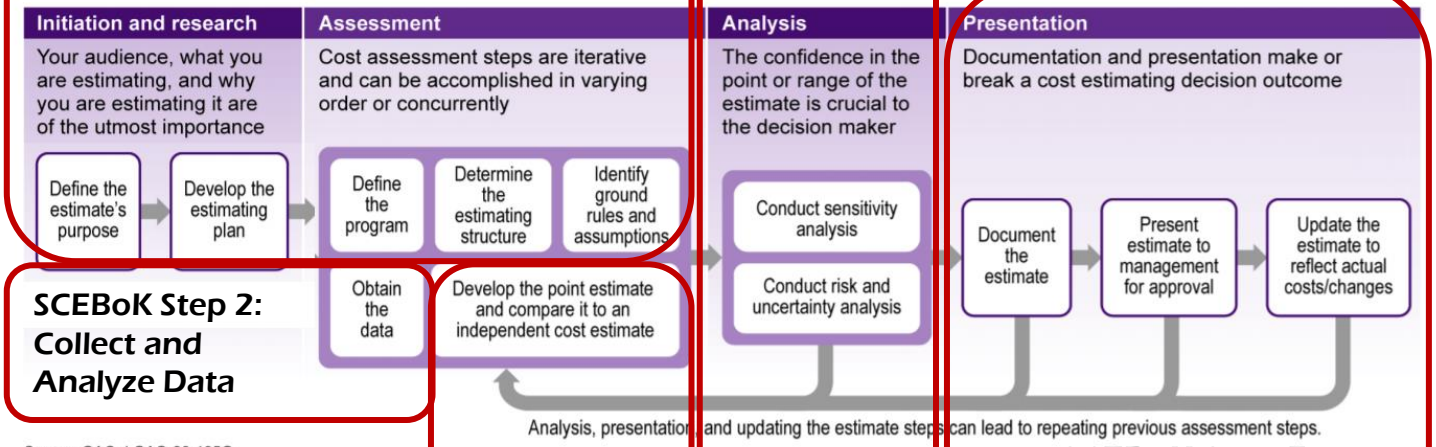
Estimates should always be expressed as a range, not an absolute.

LESSON 3: SCEBOK FIVE-STEP ESTIMATING PROCESS

Mapping the SCEBoK five-step process to the U.S. Government gold-standard process

SCEBoK Step 1: Develop Scope & Approach

SCEBoK Step 4: Conduct



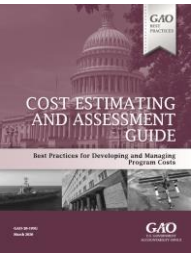
SCEBoK Step 2: Collect and Analyze Data

Source: GAO. | GAO-20-195G

SCEBoK Step 3: Create Software Estimate

Sensitivity, Risk, & Uncertainty Analysis

SCEBoK Step 5: Document and Present Estimate



Source: U.S. Government Accountability Office (GAO) COST ESTIMATING AND ASSESSMENT GUIDE: Best Practices for Developing and Managing Program Costs (GAO-20-195G), March, 2020.

Backup: GAO Positioning in the US Federal Government

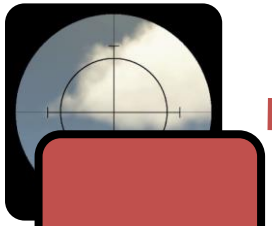
SCEBoK

Data-based estimates (DoD CADE, ISBSG, your own data)

- Data analysis is important (data must be similar, relevant, comparable)
- Data normalization is critical (units of measure, scope, who, what, OT)
- Realistic, actuals of effort and schedule data tell a story (CER, SER)

LESSON 3: SCEBOK FIVE-STEP ESTIMATING PROCESS

SCEBoK Five-step Software Estimating Process

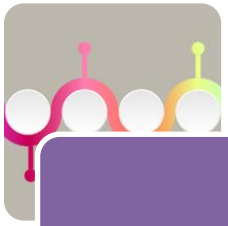


Step 1:

a. Purpose & scope
b. Develop a plan
c. Develop CES
d. Understand Technical baseline
e. Document Ground Rules & Assumptions (GR&A)

Step 2:

a. Collect,
b. Analyze, and
c. Normalize comparable historical data



Step 3:

a. Select technique(s)
b. Create point estimate
c. Perform cross-checks

Step 4:

a. Conduct sensitivity analysis
b. Conduct risk & uncertainty CRUA analysis

Step 5:

a. Document the estimate
b. Present to stakeholders
c. Update as needed

SCEBoK

LESSON 4:
ESTIMATING
CUSTOM
SOFTWARE
DEVELOPMENT

Reviewing Estimates Prepared by Others

When reviewing estimates prepared by others, look for answers:

- Are the estimated costs and schedule consistent with demonstrated accomplishments on other projects?
- Did the estimate follow a structured and documented process for relating estimates to actual costs and schedules of completed work?
- Do cost and schedule estimates quantify demonstrated organizational performance in ways that normalize for differences among software products and projects? (Note: it should not be a simple, unnormalized, lines-of-code per staff-month extrapolation used as the basis for the estimate)?
- Did extrapolations from past projects account for differences in application technology and the effects of introduction of new software technology of processes?
- Did extrapolations from past projects account for observed, long-term trends in software technology improvement?

SCEBoK