cobec
CONSULTING

QUANTIFYING THE FUTURE

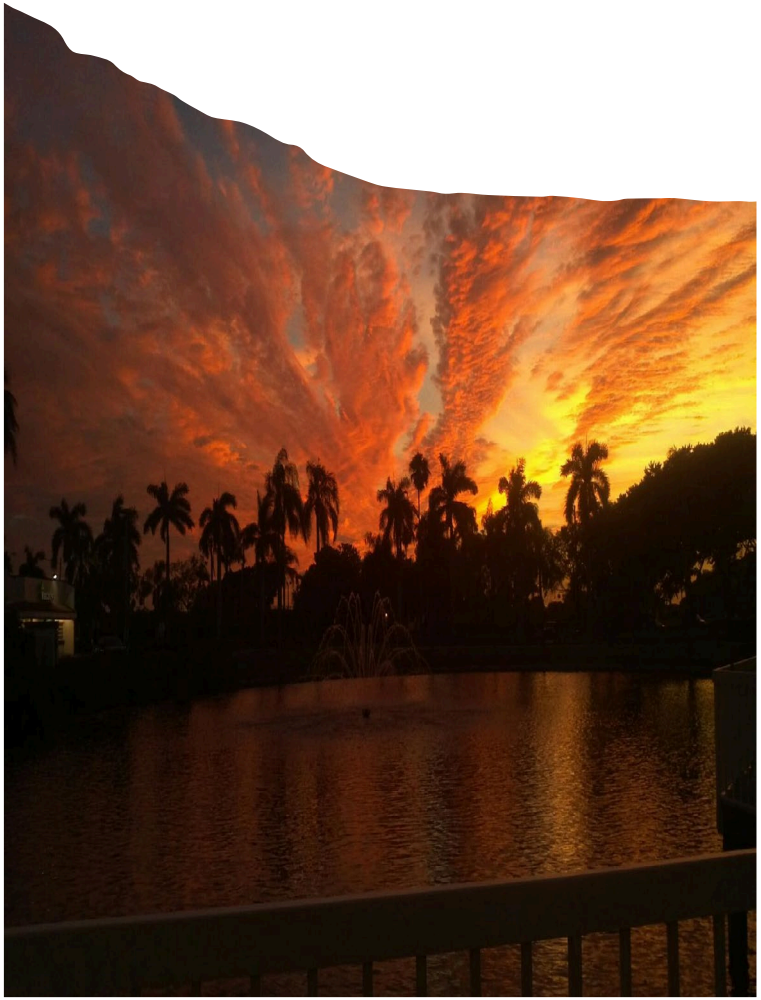
Quality Plus Technologies, Inc.

From Software ConOps to ROM in Six Easy Steps (PB04)

Carol Dekkers, CFPS (Fellow), PMP, CSM
President, Quality Plus Technologies, Inc
Office: 813-816-1329
Dekkers@qualityplustech.com

Dan French, CFPS, PMP, CSM
Domain Expert, COBEC
Office: 571-225-03
dfrench@cobec.com

Highlights - Carol Dekkers, PMP, CFPS (Fellow), P.Eng. CSM



Proud moments: ICEAA 2022 Educator of the year & Lead Author of CEBOK-S, 2023 Global Leader in Consulting, IFPUG Honorary Fellow

U.S. expert and project editor for ISO/IEC JTC1 SC7 SW Engineering standards. IFPUG Past President & current Industry Standards Committee Chair

Published author, speaker (30+ countries),consultant

Mother of 2, YaYa of 1, event & volunteer coordinator, FL resident, and a passion for tennis, travel, craft beverages and gourmet food

Highlights - Dan French, PMP, CFPS (Fellow), CSM

- B.S. in Economics from Virginia Tech
- Graduate of the Chubb Institute Top Gun Program
- Over 20 years experience in software cost estimation
- Counting function points for 24 years and been a Certified Function Point Specialist (CFPS) for 22 years (IFPUG Fellow)
- Experience in a number of estimation techniques and tools including SEER-SEM, COCOMO, SLiM, Delphi, and Estimating by Analogy
- Certification Chair for the International Function Point Users Group (IFPUG)
- Recent Certification Director for the IFPUG Board of Directors
- Former Chairman of the IFPUG Functional Software Sizing Committee (FSSC)
- GAO Agile and Cost guides expert team member
- Project Management Institute (PMI) Project Management Professional (PMP)
- Agile Alliance Certified SCRUM Master (CSM)

Topics

- Software Development Cost Drivers
  - Software size
  - Productivity
- ConOps to ROM in 6 Easy Steps
  - History of Function Points (IFPUG and Simple FP)
  - Terminology
- Case Study
- Conclusion

Software Development Cost Drivers

CEBoK-S: Lesson 3: Cost Drivers

Cost drivers should be evaluated for use as explanatory variables in creating either an analogy (e.g., effort per size measure) or parametric, regression-based relationship where effort is estimated as a function of one or more explanatory variables (e.g., size, productivity)

a. Software Size[1]

- Size of the development effort
- An important cost driver
- Key measures of size:
  - **Physical size**
  - **Functional, and non-functional size**
  - **Relative effort size**
  - **Others (requirements, RICEFW[2])**

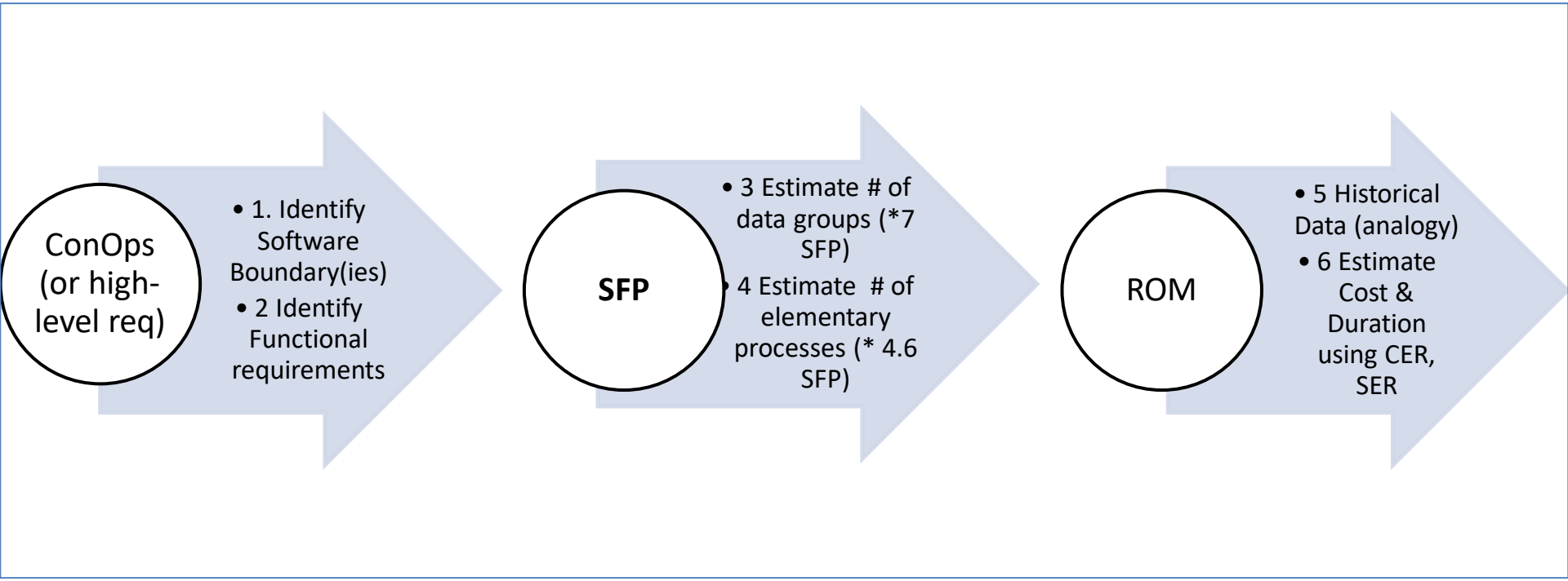
b. Productivity[3]

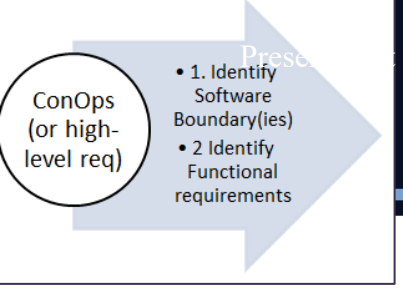
- Represents the speed at which software can be developed, … often output size /input effort
- Productivity factors often based on:
  - **Software complexity**
  - **Development team capability**
  - **Schedule (duration constraints)**

1. *Software size is the subject of separate Lesson X*
2. *RICEFW is Reports, Interfaces, Conversions, Extensions, Forms and Workflows, which are objects used to size procured software (see Lesson 6)*
3. *Productivity is the topic of separate Lesson Y*

ConOps to ROM in 6 Easy Steps

ConOps to ROM 6 steps

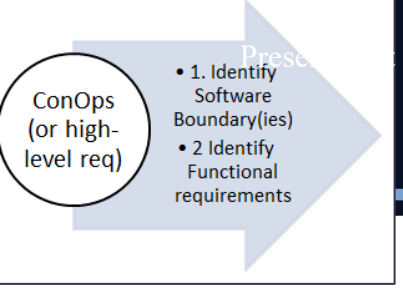




Concept of Operations (ConOps)/2

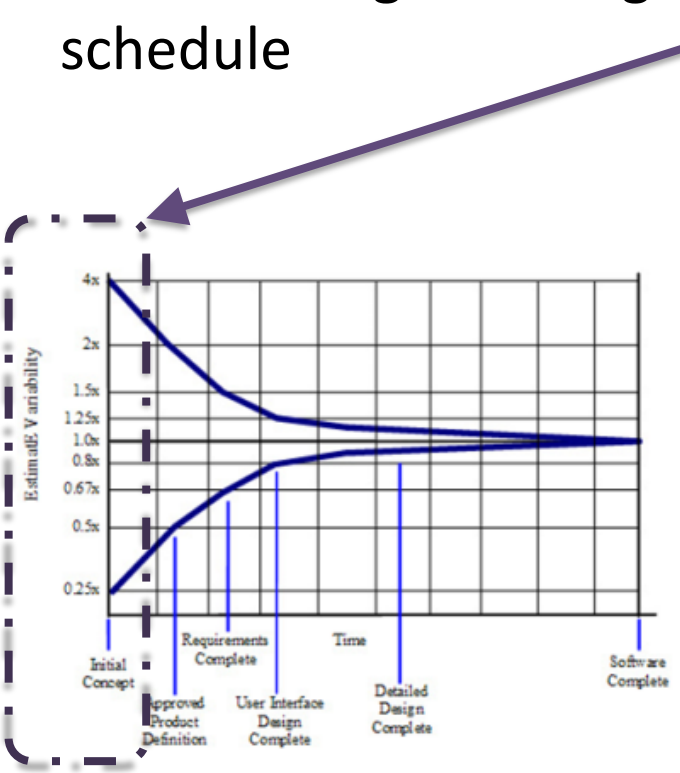
- ConOps is a **conceptual description of the operational part of the system solution.**
  The operational part of the system solution is that part of the solution which is intended to meet the requirements on the system which serve an end-use purpose.
  Source: https://www.ppi-int.com/resources/systems-engineering-faq/what-is-the-difference-between-an-ocd-conops/

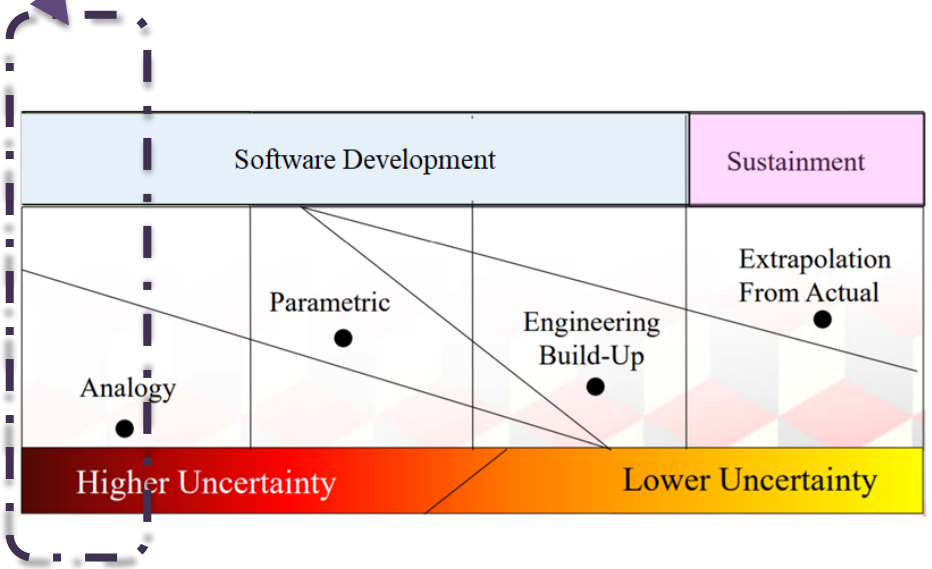
Concept of Operations (ConOps) for Program xxx



Concept of Operations (ConOps)

- A ConOps document contains **high-level software requirements**
- Suitable for generating a **ROM** estimate for size, effort, cost, schedule



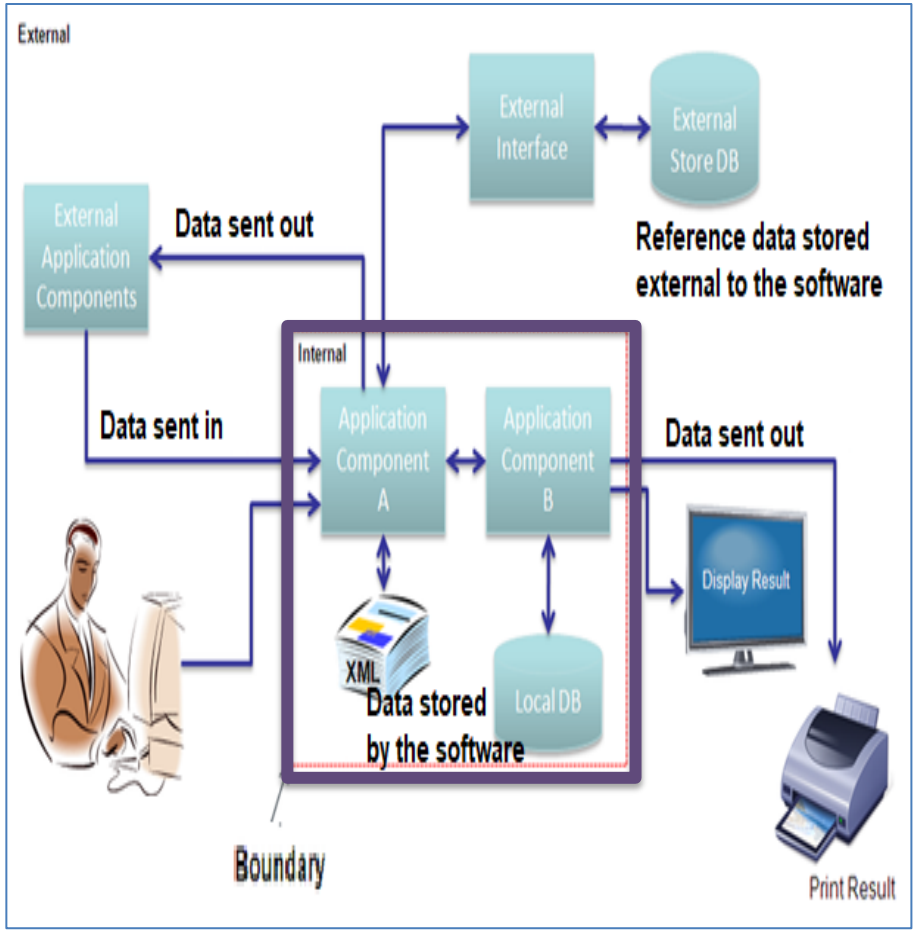


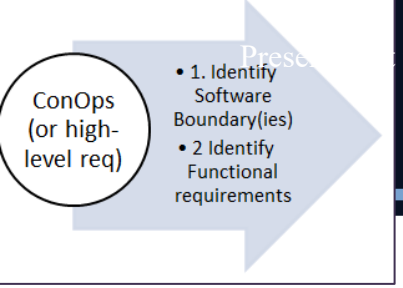
ConOps (or high-level req)

- 1. Identify Software Boundary(ies)
- 2 Identify Functional requirements

Step 1: Identify Software Boundary(ies)

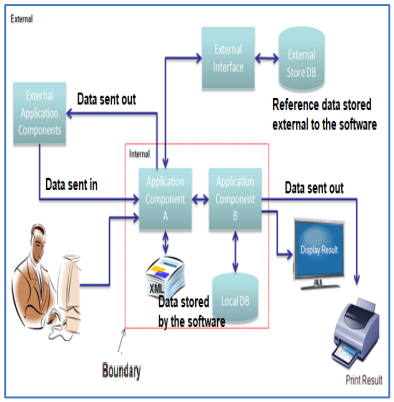
- Conceptual line between the software under analysis and its users, through which data passes into and out of the application(s)
- Most crucial step in the sizing process
- ConOps could contain MULTIPLE pieces of software → separate sizes





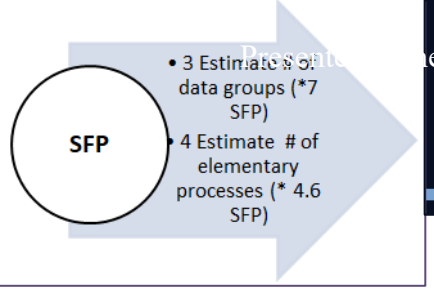
Step 2: Identify Functional (User) Requirements (FUR)

- A sub-set of the user requirements; requirements that describe **what the software shall do**, in terms of tasks and services.
- Identify logical groups of data and processes that describe:
  - Data to be referenced or stored
  - Reports to be produced
  - Displays of data
  - Send of data to other systems (interface or output)
  - Data entry
- Exclude non-functional (e.g., performance, quality), and technical requirements*

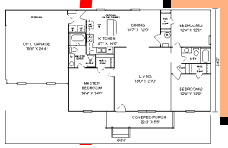


* Non-functional requirements can be estimated using FPUG Software Non-Functional Assessment Points (SNAP) or other approaches

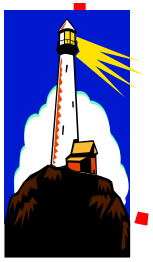
Function Points measure the size of Functional User Requirements



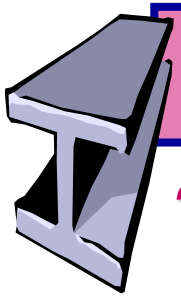
"User Requirements"



1. Functional (User) Requirements (WHAT: FP)



2. Non-Functional Requirements (HOW GOOD)

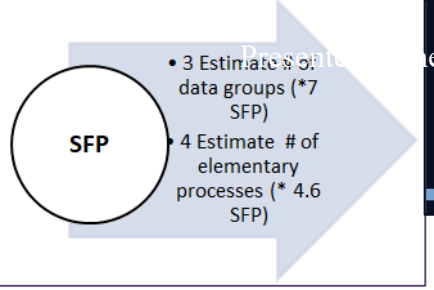


3. Technical Requirements (HOW TO BUILD)

"Developer Construction Requirements"

Software Requirements

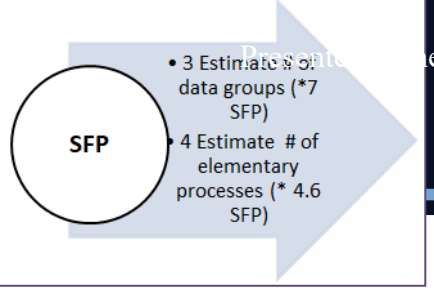
Note: Effort, Cost, & Schedule are a function of all types



History of Function Points (FP)

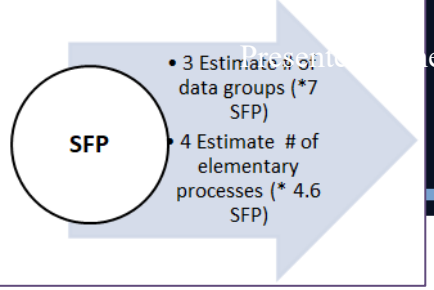
- **Mid-1970's:** Function Points (FP) developed at IBM as an alterative to Source Lines of Code (SLOC)
- **1984-1986:** Formation of the International Function Point Users Group (IFPUG) and publication of IFPUG FP Counting Practices Manual v1.0
- **1998:** ISO/IEC 20926 (IFPUG FP) and other ISO standards for Functional Size
- **2010:** IFPUG 4.3.1 - current version. Assigns FP to 5 function types and 3 complexities (low, avg, high) based on detailed software requirements
- **2021: IFPUG Simple FP (SFP) v2.1** – Assigns SFP to 2 function types (one complexity)

IFPUG International Function Point USERS GROUP



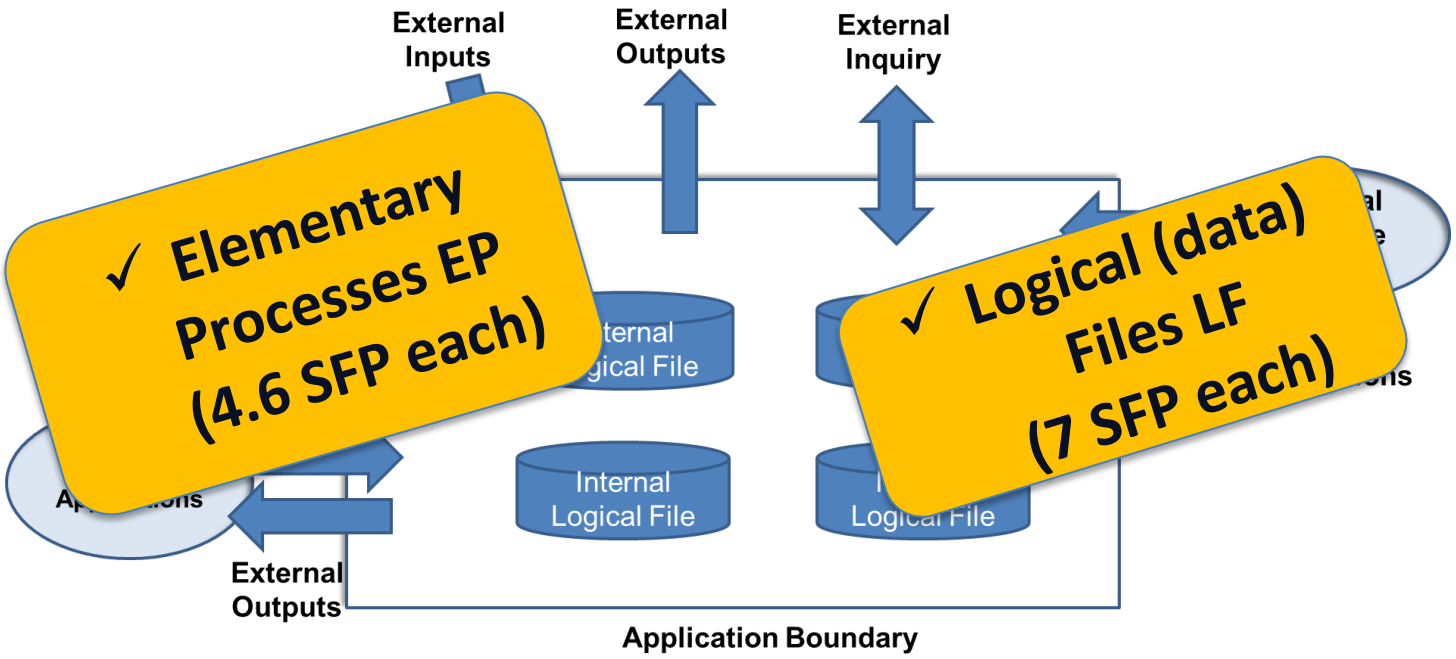
IFPUG SFP: Simple Function Points

- IFPUG SFP v2.1 (2021) is a simplified (and standardized) approach compatible with IFPUG v4.3.1
- Size is based on two functional components (single complexity):
  - Logical files (Data Groups) → 7 SFP each
  - Elementary Processes →4.6 SFP each
- Especially suitable for early software sizing when details of FUR are not yet known

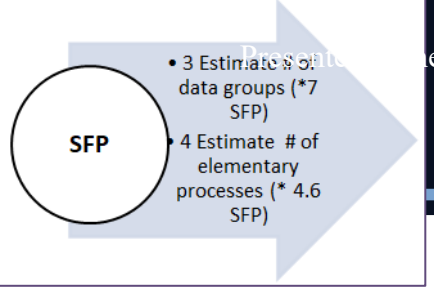


IFPUG FP v4.3.1 versus IFPUG SFP v2.1

✓ IFPUG Simple FP v2.1



To "estimate" IFPUG SFP → Identify Logical Files & Elementary Processes → Translate # FP

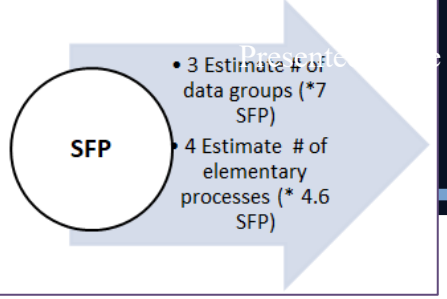


Step 3: Estimate Logical Files (Data Groups)/2

Logical file (LF)

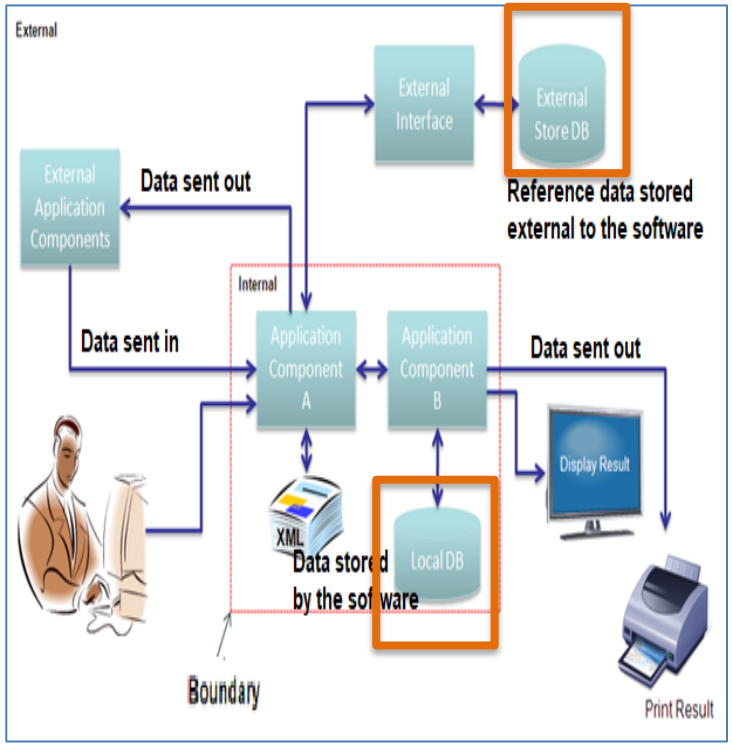
- functionality provided to the user to meet **internal and external data storage requirements <Data group>** I
- user recognizable group of logically related data or control information maintained and/or referred within the boundary of the application being measured."
- The term file here does not mean physical file or table. In this case, file refers to a logically related group of data and not the physical implementation of those groups of data.

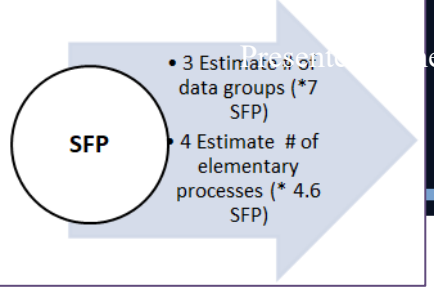
Each LF = 7 SFP



Step 3: Estimate Logical Files (Data Groups)

- Does not matter if internal (maintained) or external (referenced)
- Identify Logical Files (data groupings)
- Estimate # of Logical Files (* 7 SFP)





Step 4: Estimate Elementary Processes /2

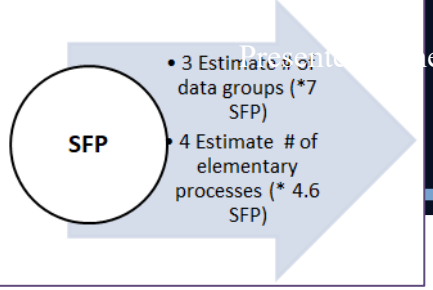
Elementary process (EP) –

- **smallest unit of activity, which is meaningful to the user, that constitutes a complete transaction,** it is self-contained and leaves the business of the application being measured in a consistent state

Examples are :

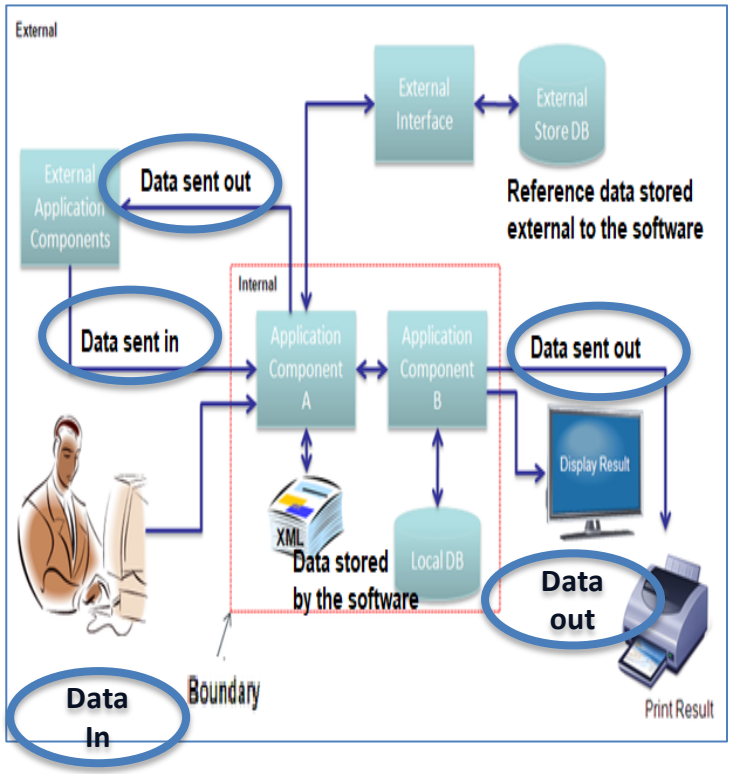
- CRUD (Create, Read, Update, Delete) = 4 Elementary Processes
- Report = 1 elementary process
- Display data = 1 elementary process

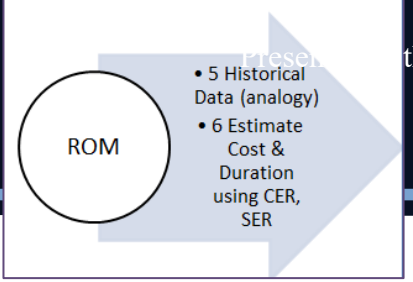
Each EP = 4.6 SFP



Step 4: Estimate Elementary Processes

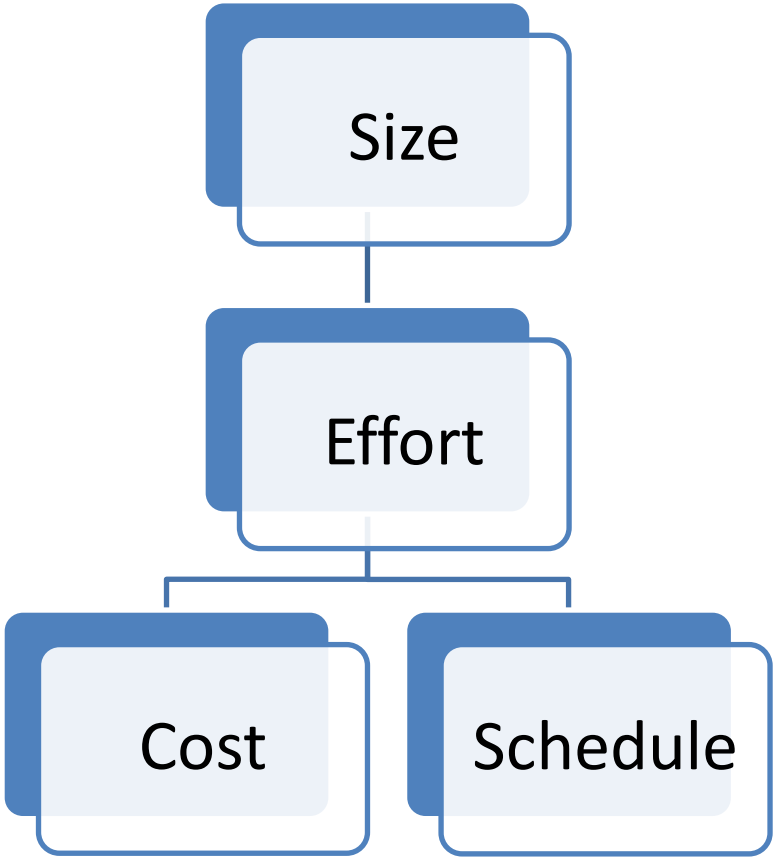
- All types of transactional functions (inputs, outputs, queries) have same SFP value
- Identify elementary processes
- Estimate # of Elementary processes (* 4.6 SFP)

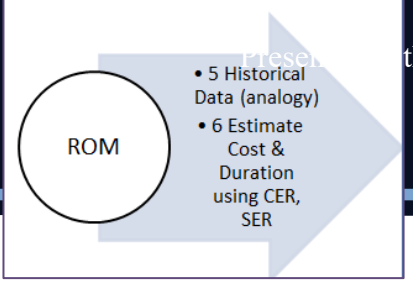




ROM (Estimate)

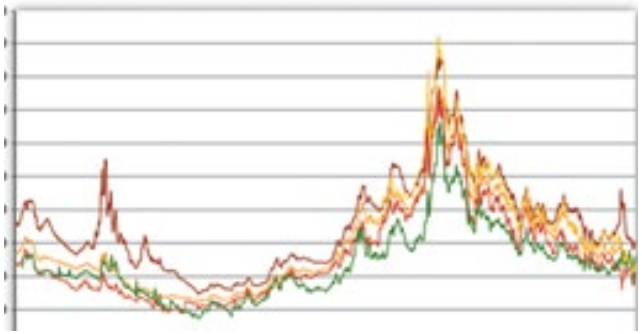
- Estimated software size (sum)
  - SFP logical files
  - SFP elementary processes
- Estimated effort
- Estimated cost
- Estimated schedule (duration)



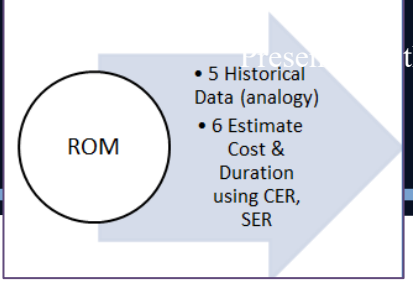


Step 5: Historical data (analogy)

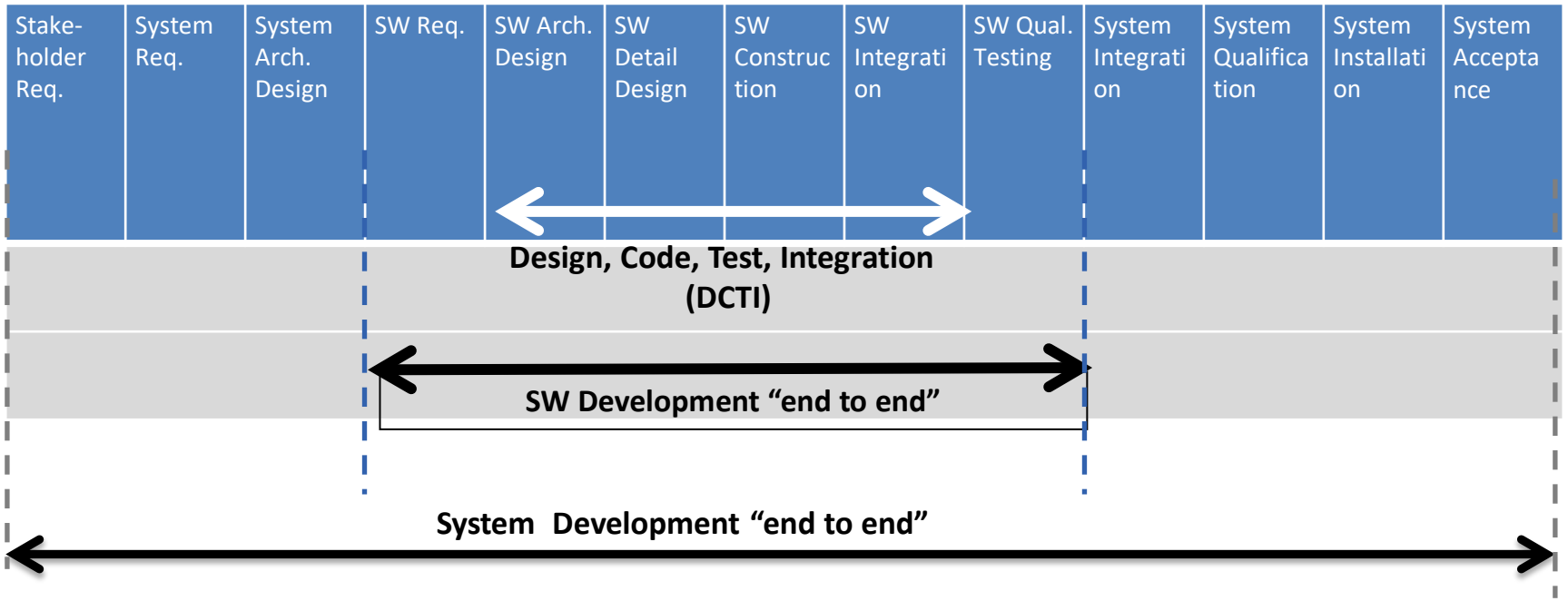
- Find similar projects in historical data
  - Similar size
  - Similar productivity factors:
    - Complexity
    - Developer capability
    - Schedule compression (if applicable)
  - May need to adjust for scope

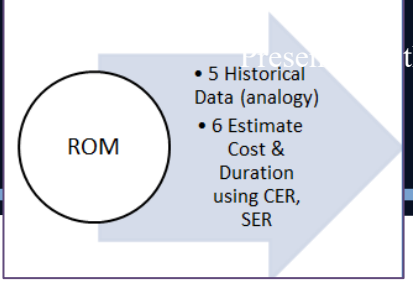


- Industry data sources:
  - SRDR (US DoD)
  - International Software Benchmark Standards Group (ISBGS) Application Development & Enhancement repository (www.isbsg.org)
  - Commercial tool datasets



Historical data
Match / adjust scope of activities

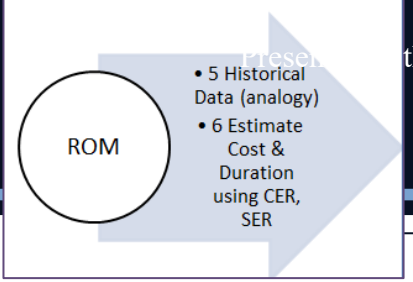




ISBSG D&E Repository

- In lieu of historical data, can use the ISBSG Development & Enhancement (D&E) Repository (2020)[1]
- Typical database filters for selecting analogous projects
  - **Data Quality Rating → A or B**
  - **Size (range close to your SFP estimate)**
  - **Year of Project**
  - **Industry Sector and Organization Type**
  - **Application Group and Application Type.**
  - **Development Type.** New development or enhancement.
  - **Count approach.→ IFPUG 4.0 and above**

1. ISBSG Development and Enhancement (D&E) Repository, Corporate Release 2020 R1, August 2020, with 9,592 completed projects



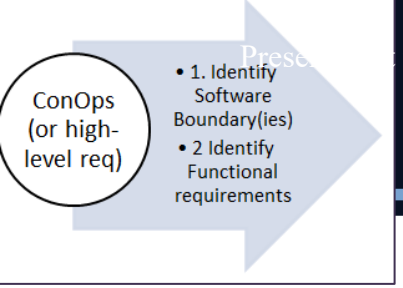
Step 6: ROM Estimate using Analogy

- Identify any applicable CERs/SERs (linear analogy):
  - Estimated effort (hours) = SFP Size * ISBSG analogy effort (hours)/ ISBSG analogy size
  - Cost = Estimated effort (hours) * labor rate per hour
  - Duration (months) = Estimated effort (hours) / (hours/PM * team size)
- Cross check using COCOMO II or commercial tool or other estimating methods
- Typically, acceptable variance range between estimates should be between 10 – 20% (maximum)

ConOps Case Study

cobec CONSULTING

Quality Plus Technologies, Inc.

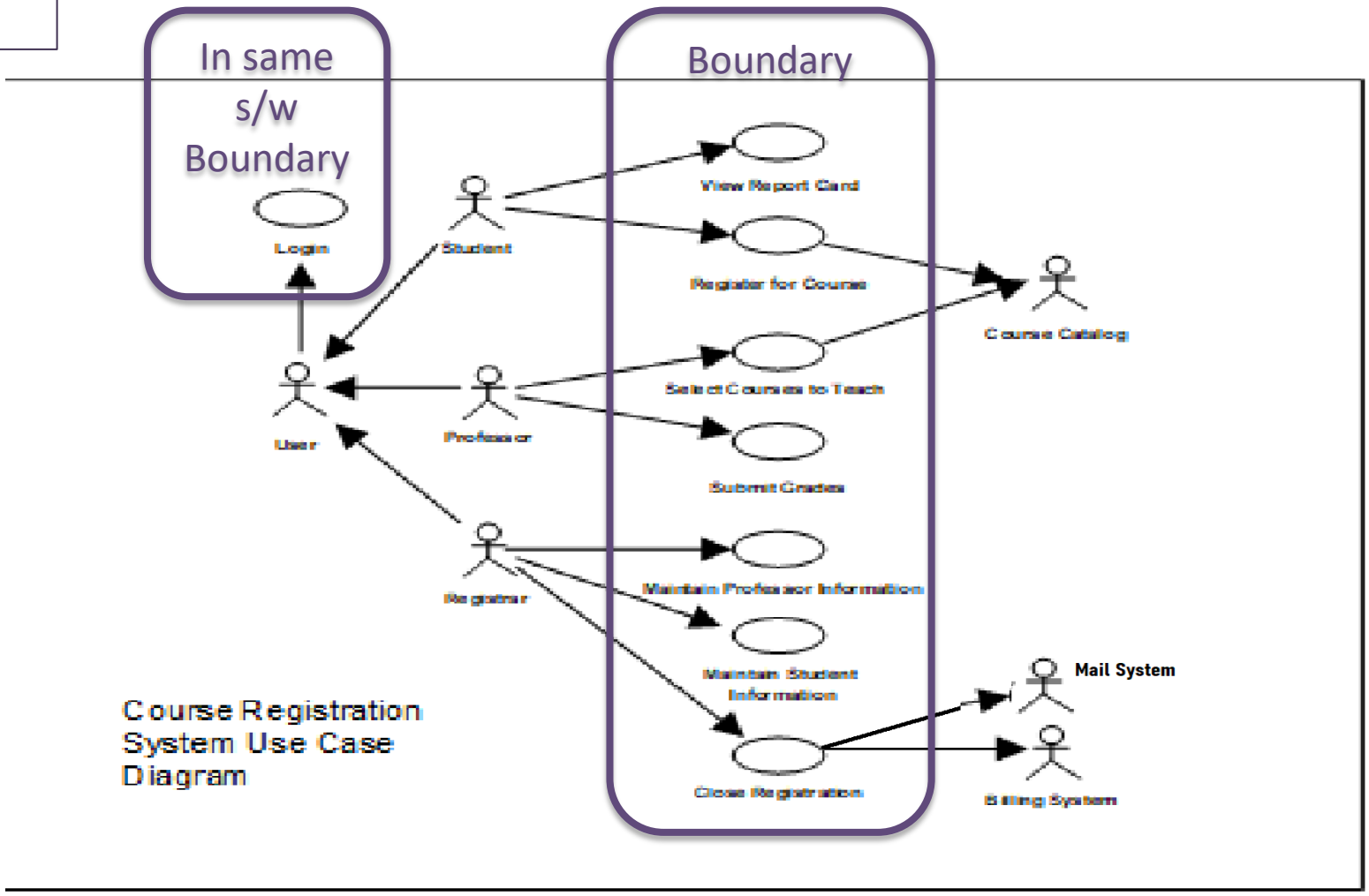


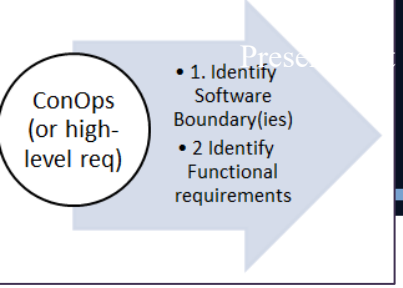
Concept of Operations (ConOps)

- Adapted from CEBoK-S Lesson X (Software Size)
- High level use case list & diagram for Course Registration System – without details
- Software will provide functions needed to maintain professor, student, and course information as well as class registration functions

- **Note: used only high-level diagram and description (CEBoK-S case study is more detailed → Different estimated size)**

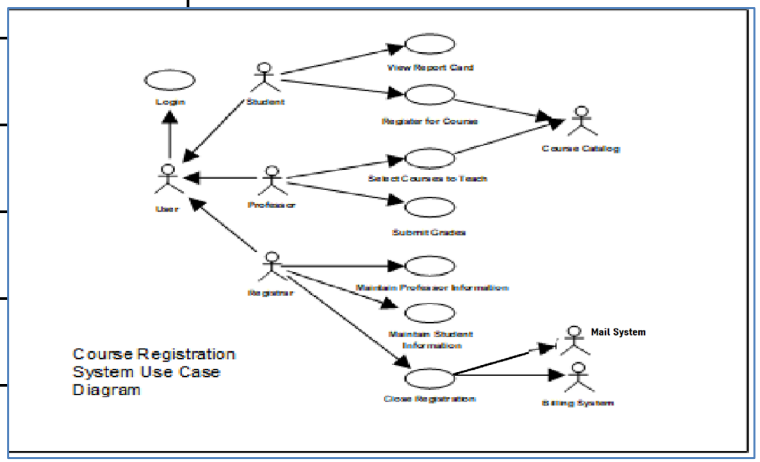
Step 1: Identify Software Boundary(ies) with Use Case Diagram

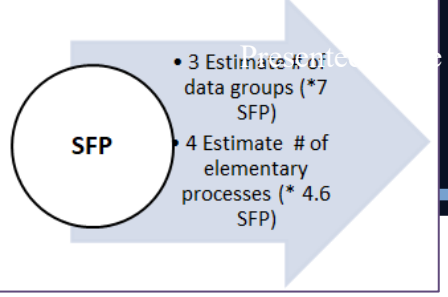




Step 2: Identify Functional (User) Requirements (FUR)

#	**Use case for Course Registration System**
1.	Logon (by all users)
2.	Maintain professor information (by the registrar)
3.	Select courses to teach (by professors)
4.	Maintain student information (by the registrar)
5.	Register for course(s) (by students)
6.	Close registration (by the registrar)
7.	Submit grades (by professors)
8.	View report card (by students)





Step 3: Estimate Logical Files (Data Groups)

#	Use case for Course Registration System
1.	Logon (by all users)
2.	Maintain professor information (by the registrar)
3.	Select courses to teach (by professors)
4.	Maintain student information (by the registrar)
5.	Register for course(s) (by students)
6.	Close registration (by the registrar)
7.	Submit grades (by professors)
8.	View report card (by students)

User logical file

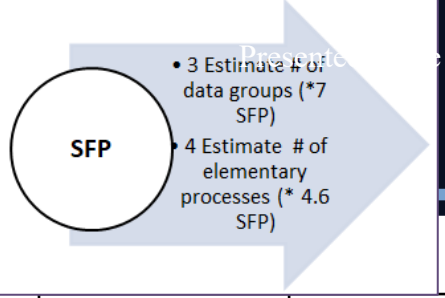
Professor logical file

Course catalog

Student information

Course registration/register includes grades

Estimated 5 Logical Files = 35 SFP



Step 4: Estimate Elementary Processes

#	Use case for Course Registration System
1.	Logon (by all users)
2.	Maintain professor information (by the registrar)
3.	Select courses to teach (by professors)
4.	Maintain student information (by the registrar)
5.	Register for course(s) (by students)
6.	Close registration (by the registrar)
7.	Submit grades (by professors)
8.	View report card (by students)

Logon = 1 EP

CRUD = 4 EP

Select= 1 EP

CRUD = 4 EP

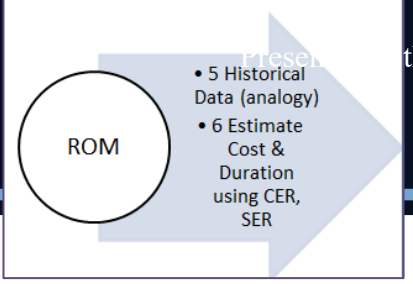
Register= 1 EP

Close = 1 EP

Submit = 1 EP

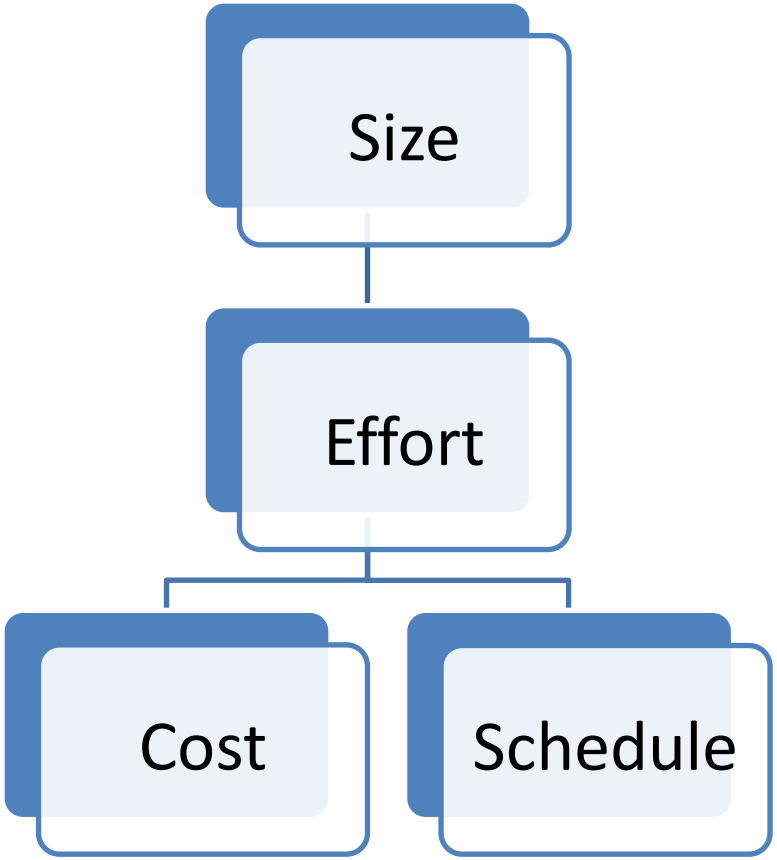
Report card output = 1 EP

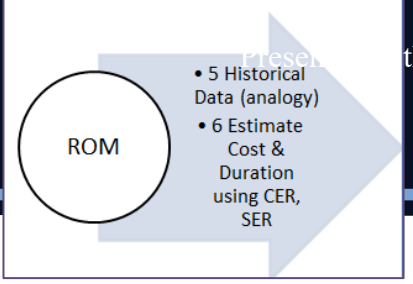
Estimated 14 EP = 64.4 SFP



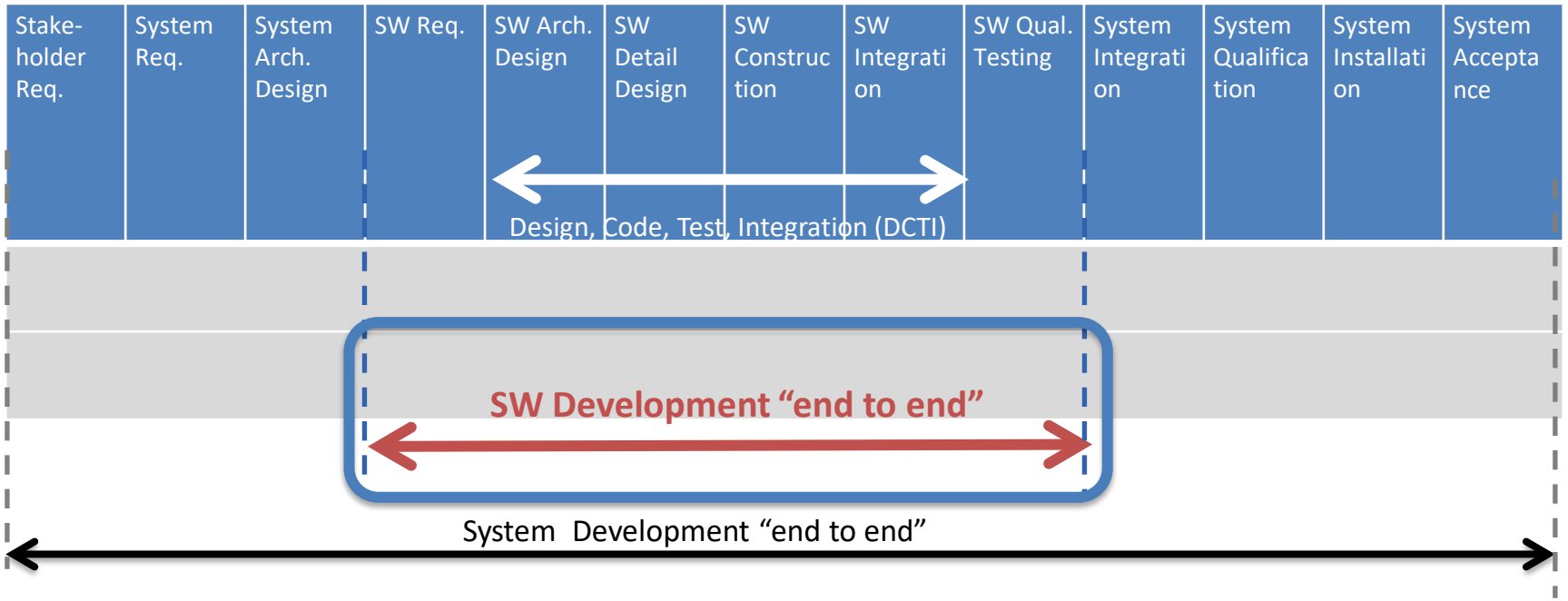
ROM Software Estimate

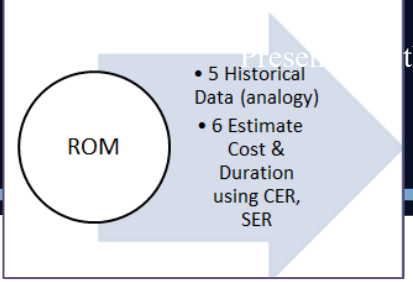
- Estimated size = 100 SFP
  - SFP logical files = 35 SFP
  - SFP elementary processes = 64.4 SFP
- Estimated effort
- Estimated cost
- Estimated schedule (duration)





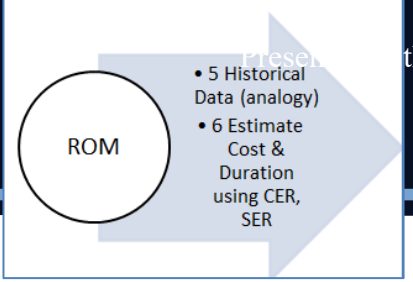
Step 5: Historical data
Scope of activities





Step 5: Historical data

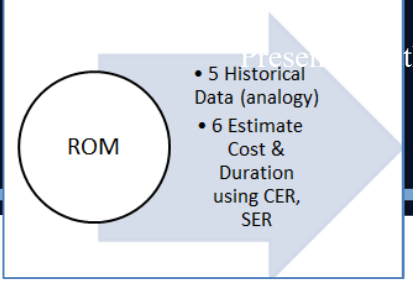
- Use organizational data if available otherwise, obtain industry data from other sources
- Key to successful Analogous estimating is to find project as similar to the project being estimated and adjust for any factors that may result in different cost & schedule
- If multiple projects exist may want to develop a composite for the analogous estimate.



ISBSG D&E Repository

- In lieu of historical data, can use the ISBSG Development & Enhancement (D&E) Repository (2020)[1]
- **We have 2007 version of ISBSG D&E (for DEMO)**
- Typical database filters for selecting analogous projects
  - **Data Quality Rating = A or B**
  - **Size = 75-200 FP**
  - **Industry Sector and Organization Type**
  - **Application Group and Application Type.**
  - **Development Type. New development.**
  - **Count approach. Select IFPUG 4.0 and higher**

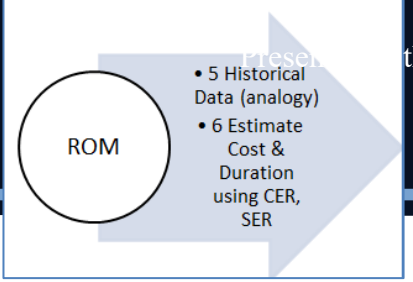
1. ISBSG Development and Enhancement (D&E) Repository, Corporate Release 2020 R1, August 2020, with 9,592 completed projects



ISBSG D&E Repository

Filter the ISBSG D&E records

Attribute	Student Registration	Mapping Required / Criteria for Inclusion
Data Quality	**A or B**	Include data points that have all or a majority of fields provided for the project – that is data quality = A or B only
Size	**75-200 FP**	Limited by ISBSG database version… >2000
Industry Sector, Organization Type	**Any sector and type**	Not used as an initial filter, unless we need to further refine our dataset
Application Group and Application Type	**Not particular**	Include only data points that map to the Application Group: Business Application, and Application Type: Application software or Financial
Development Type	**New Development**	Include only data points that are new development
FP Counting approach	**IFPUG 4+**	Include only data points that have sizing units of measure IFPUG 4.0 or newer (compatible with current IFPUG 4.3.1)



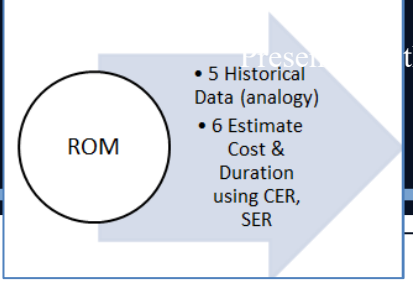
ISBSG D&E Repository

After filtering based on our criteria, then normalizing (as necessary) the activities and effort, our dataset now has 6 potentially analogous records that match

	Rating	Sizing		Effort		oductiv	Schedule			
	Data Quali	Count	Functional Size	Normalised Work Effort	Summar	Normalised PDR (ufp)	Project Elapsed Time	Project Activity Scope	Develop	Organisa
	A	IFPUG	149	193	174	1.3	2	Specification;Build;Test;Imp	New Dev	Public Ad
	B	IFPUG	106	530	350	5.0	2	Planning;Specification;Build	New Dev	Public Ad
1	B	IFPUG	188	4599	4599	24.5	9	Planning;Specification;Build	New Dev	Manufact
	A	IFPUG	98	487	438	5.0	1	Specification;Build;Test;Imp	New Dev	Public Ad
2	A	IFPUG	103	1807	1626	17.5	4	Specification;Build;Test;Imp	New Dev	Public Ad
	B	IFPUG	125	304	170	2.4	6	Planning;Build;	New Dev	Public Ad

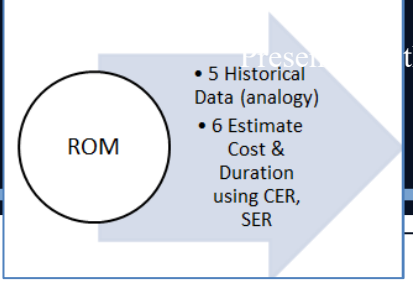
We want to further refine the selection to find **the most analogous project**. We selected and highlighted **two projects (see 1 and 2)** **as our potential best-fit analogous projects. (#2 would need to normalize for planning)**

Note: an alternate approach would be to use these 6 records as the basis for a parametric CER. See CEBoK-S lesson 4 for details.



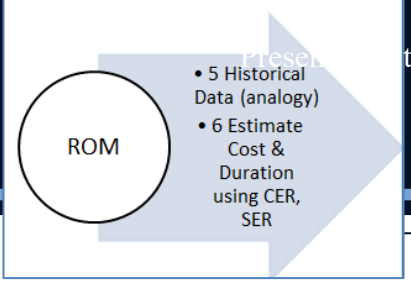
Step 6: ROM Estimate using Analogy

- Identify any applicable CERs/SERs (linear analogy):
  - Est Effort (hours) = SFP Size * ISBSG analogy effort (hours) / ISBSG analogy size
    - Est Effort (hours) = 100 SFP * 4599 hours / 188 FP = 2446 hours
    - At 152 hour/PM = 16 PM
  - Cost = Estimated effort (hours) * labor rate per hour
    - Assumed labor rate = $60 USD / hour
    - Cost = $146,760
  - Duration (months) = Estimated effort (hours) / (hours/PM * team size)
    - Duration (months) = 2446 hours / (152 hours/PM *2 people) = 8 months
    - ISBSG project elapsed time = 9 months (see previous page)



Step 6: ROM Estimate using Analogy

- Can include data for Effort per SFP, SPF/Mo, $/SFP
- Based on the adjustments made to the historical data, CERs, and SERs, the estimate can be developed.
- Cross check using COCOMO II or commercial tool or other estimating methods
- Typically, acceptable variance range between estimates should be between 10 – 20% (maximum)



Cross-checking the ROM Estimate

Preliminary cross check using COCOMO II™ web tool

http://softwarecost.org/tools/COCOMO/

Created by Ray Madachy at the Naval Postgraduate School. Email: rjmadach@nps.edu.

COCOMO II - Constructive Cost Model

Software Size Sizing Method: Function Points

Unadjusted Function Points: 100 Language: Java

Software Size Probability Distribution

Iterations: 0 0 0 0 0 0

0-0 0-0 0-0 0-0 0-0 0-0

Software Equivalent Size (KSLOC)

Distribution Type: Normal

Iterations: 1000

Software Scale Drivers

Precedentedness	Nominal	Architecture / Risk Resolution	Nominal	Process Maturity	Nominal
Development Flexibility	Nominal	Team Cohesion	Nominal		

Software Cost Drivers

Product		Personnel		Platform	
Required Software Reliability	Nominal	Analyst Capability	Nominal	Time Constraint	Nominal
Data Base Size	Nominal	Programmer Capability	Nominal	Storage Constraint	Nominal
Product Complexity	Nominal	Personnel Continuity	Nominal	Platform Volatility	Nominal
Developed for Reusability	Nominal	Application Experience	Nominal	**Project**	
Documentation Match to Lifecycle Needs	Nominal	Platform Experience	Nominal	Use of Software Tools	Nominal
		Language and Toolset Experience	Nominal	Multisite Development	Nominal
				Required Development Schedule	Nominal

Maintenance Off

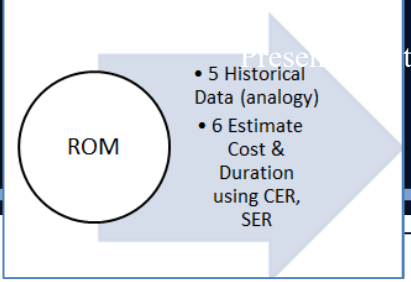
Software Labor Rates

Cost per Person-Month (Dollars): 9120

Calculate

Assumptions:

- 100 FP (Selected Java)
- Labor rate = $9120/PM (152 hours/PM * $60/hour)
- All nominal settings



Cross-checking the ROM Estimate

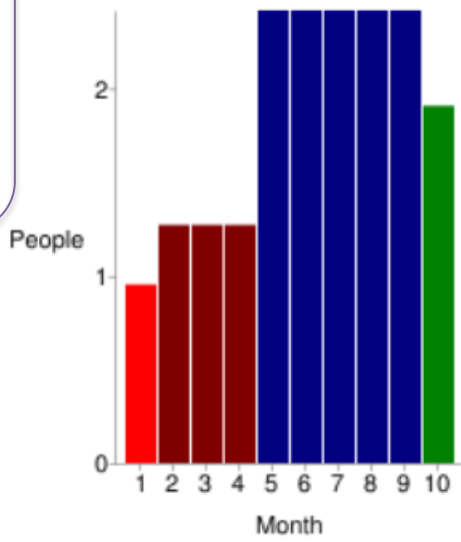
Results

Software Development (Elaboration and Construction)

Effort = 18.4 Person-months
Schedule = 9.3 Months
Cost = $167814

Total Equivalent Size = 5300 SLOC
Effort Adjustment Factor (EAF) = 1.00

Staffing Profile



Acquisition Phase Distribution

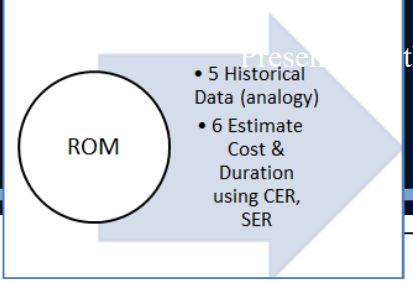
Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.1	1.2	1.0	$10069
Elaboration	4.4	3.5	1.3	$40275
Construction	14.0	5.8	2.4	$127539
Transition	2.2	1.2	1.9	$20138

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.2	0.5	1.4	0.3
Environment/CM	0.1	0.4	0.7	0.1
Requirements	0.4	0.8	1.1	0.1
Design	0.2	1.6	2.2	0.1
Implementation	0.1	0.6	4.8	0.4
Assessment	0.1	0.4	3.4	0.5
Deployment	0.0	0.1	0.4	0.7

Results:

- Est Effort (Elaboration + Construction) = 18.4 PM *152 hours/PM = 2798 hours
- Est Cost = $167,814
- Est Schedule = 9.3 months



ROM Estimate(s)

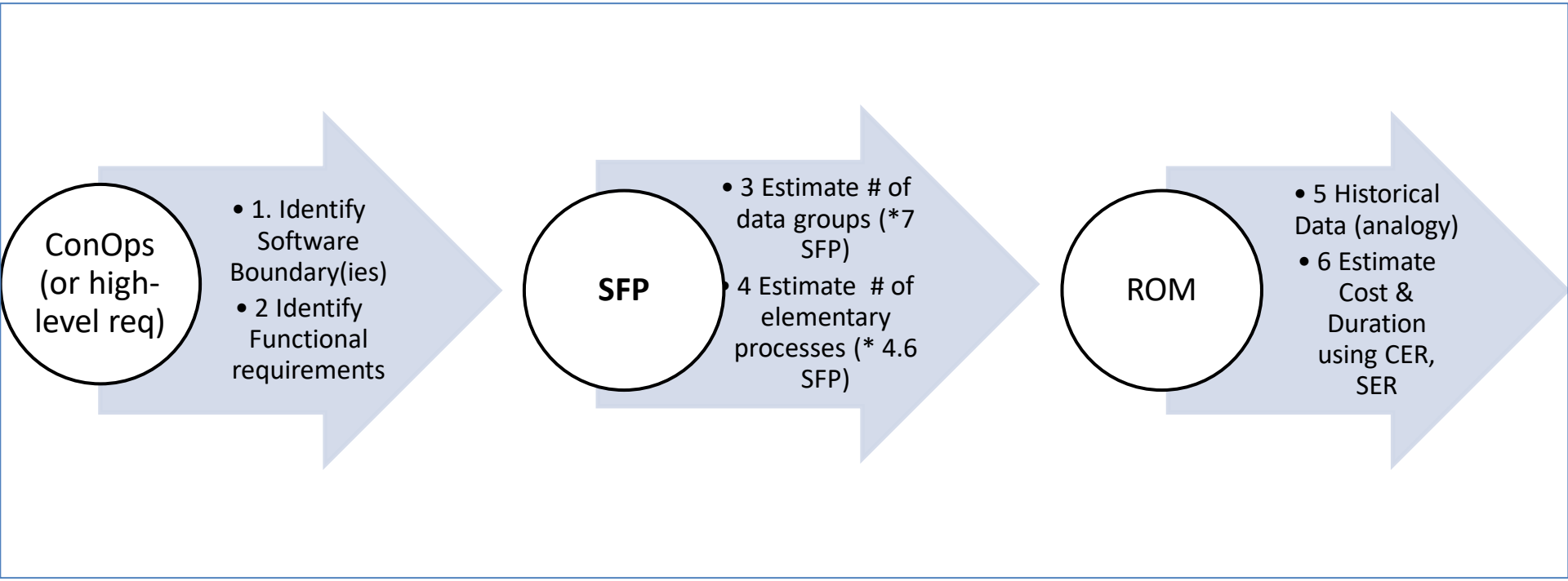
ROM (based on Est Size = 100 SFP)	Analogy Estimate	COCOMO II Cross Check
Estimated Effort (hours) (PM)	2446 hours 16 PM	2798 hours 18.4 PM
Estimated Cost ($)	$147 K USD	$167 K USD
Estimated Schedule (months)	9 months	9.3 months

ROM Estimate Cross-check

- Proof of concept (process)
- Historical data and CER/SER available:
  - DHS CAD (Dr. Wilson Rosa, Sara Jardine)
  - ISBSG D&E repository 2022 (much larger database)
  - Commercial tools
- Simple Function Points (SFP) is compatible with IFPUG FP

ConOps to ROM in 6 Easy Steps

A viable approach



Conclusions

- IFPUG function point methodology is a tried-and-true, ISO/IEC standardized method for software sizing
- Early size estimates are possible from ConOps and high-level requirements using IFPUG Simple function points (SFP)
- SFP provide a simpler way to size FURs (with reasonable accuracy) especially on high-level ConOps or EPICS or user stories when cost analysts are not trained/certified
- Start collecting <good> historical data:
  - Actual IFPUG FP counts (delivered software)
  - Actual software development effort, cost and schedule
  - Ensure you record Productivity Factors and Contextual data

Resources

- International Cost Estimating and Analysis Association (ICEAA) https://www.iceaaonline.com/ -
  - CEBoK-S Lesson X: Software Size includes authoritative software sizing (and full case study using multiple functional sizing methods)
  - Extensive techniques for estimating software programs (including hybrid)
- International Function Point User Group (IFPUG) http://ifpug.org/
  - IFPUG Function Point Analysis v4.3.1
  - IFPUG Simple Function Points (SFP) v2.1
- International Software Benchmarking Standards Group (ISBSG) http://www.isbsg.org D&E and Maintenance repositories

cobec
CONSULTING

QUANTIFYING THE FUTURE

Quality Plus Technologies, Inc.

THANK YOU

Carol Dekkers, CFPS (Fellow), PMP, CSM
President, Quality Plus Technologies, Inc
Office: 813-816-1329
Dekkers@qualityplustech.com

Dan French, CFPS, PMP, CSM
Domain Expert, COBEC
Office: 571-225-03
dfrench@cobec.com

CEBoK-S Case Study – See Lesson X: Software Size

SCEBoK - Functional Size Estimation Case Study:
Course Registration System –
Illustrating *high-level, early estimating methods* for COSMIC, IFPUG, Nesma, SiFP and Use Case Points
by Carol Dekkers, PMP, CFPS (Fellow), P.Eng.

<Author notes (Carol Dekkers)>

1. *This case study was originally published in **The IT Measurement Compendium: Benchmarking and Estimating Success with Functional Size Measurement** by Manfred Bundschuh and Carol Dekkers, 2008, Springer Publications, Germany, chapter 18 as a comparison of FP counts.*
2. *For the SCEBoK, this case study is corrected updated, abridged and adapted to include a demonstration of how to ESTIMATE the functional size, using high-level early-estimating methods, rather than using the original, detailed full methods:*
   - *IFPUG 4.3.1 – high-level and indicative FPA*
   - *COSMIC v5.0 – COSMIC- Equal Size Bands*
   - *Nesma v2.3 – high-level FPA*
   - *Simple Function Points (SiFP) v1.01*
   - *Use Case Points (UCP)*
3. *Case Study results were reviewed by Functional Size Measurement experts:*
   - *IFPUG 4.3.1 high-level and indicative FPA: Dan French*
   - *COSMIC v5.0 – COSMIC Equal Size Bands: Frank Vogelzang*
   - *Nesma v2.3 – high-level FPA: Hans Bernink*
   - *Simple Function Points (SiFP) v1.01: Carol Dekkers*
   - *Use Case Points (UCP): Dr Alistair Cockburn*

Because this case study is used to illustrate the steps and various decisions for doing a particular ESTIMATE, we are not focused on the style of the use case narrative. Note that experts for the various software functional size measurement methods from COSMIC, IFPUG, and Nesma reviewed the FP estimates for each of their high-level methods, and Dr. Alistair Cockburn (originator of the initial use cases used in this case study) reviewed the section on estimated Use Case Points.

4. *While an updated, but similar, case study still exists on the COSMIC website, the details and functions have been revised and, therefore, the estimated FP commensurately do not match those presented here. Note that this case study illustrates using the high-level, estimating versions of IFPUG v4.3.1, Nesma v2.3, and COSMIC v5.0 and provides estimates of functional size, not measured counts. There is not enough information to be able to count the functional size because we do not have the information about data entities, or how the data are manipulated via transactions or elementary processes.*

History of Simple Function Points

- 2009: Dr. Roberto Meli of DPO introduces the Early & Quick Function Points (E&Q FP) based on the IFPUG method. New concepts:
  - Generic Functions
  - Typical Process (TP) (CRUD)
  - Generic Process (GP)
  - Macro Process (MP)
- 2010: Meli refined E&Q FP into Simple Function Points (SiFP) with 2 generic function types:
  - Elementary Process (EP)
  - Logical File (LF)
- 2019: IFPUG acquired the SiFP method
- 2021: IFPUG releases IFPUG Simple Function Point (SFP) manual v.2.1

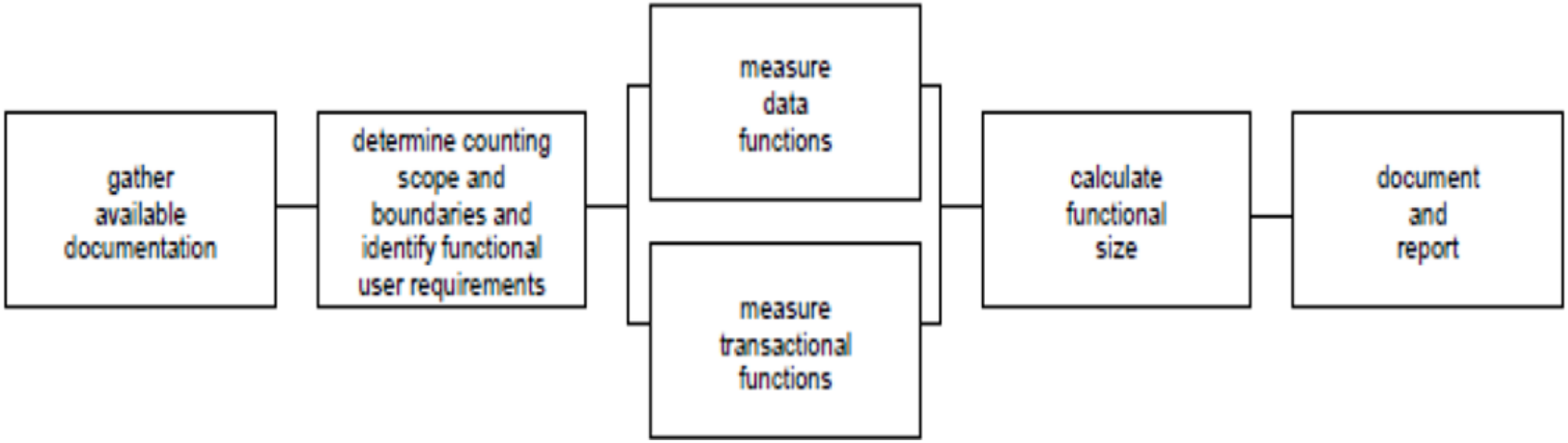
IFPUG FP vs Simple FP

- International Function Point Users Group (IFPUG) SFP v2.1 (2021)
- Originally developed by Dr. Roberto Meli/Italian researchers v1.1 (2010))
- Simplifies functional sizing into two types of functions:
  - Generic elementary processes (transactional functions)
  - Generic logical files (data groups)

IFPUG Components	Low	Average	High
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiries	3	4	6
Internal Logical Files	7	10	15
External Interface Files	5	7	10

Simple Function Point SFP components	Weighting Factor
Elementary Processes EP (Transactional Functions)	4.6 SFP
Logical Files LF (Data Functions)	7 SFP

IFPUG FP & SFP Measurement Process



Backup: IFPUG FP vs IFPUG SFP (1 of 2)

Concept	IFPUG FP	IFPUG SFP
IFPUG standardized glossary	Yes	Yes, same
Intent to measure functional size based on FUR	Yes	Yes, same
Method owned by IFPUG	Yes	Yes
IFPUG FP measurement steps: 1. Gather available documentation 2. Purpose/scope/boundary, identify FUR 3a. Measure data functions 3b. Measure transactional functions 4. Calculate functional size 5. Document and report	Yes, but steps 3a and 3b involve additional sub-steps: subclassification into 3 types of transactional functions and 2 types of data functions, and a complexity classification (into Low, Average, or High) to get FP values	Yes
Base functional components (BFC): transactional functions and data functions	Yes: Transactional functions are subdivided into EI, EO, EQ, and Data functions are subdivided into ILF, EIF	Yes: Transactional functions are called “Elementary Processes” and Data Functions are called “Logical Files”

Backup: IFPUG FP vs IFPUG SFP (2 of 2)

Number of different FP values allocated across function types	3 FP values allocated as Low, Average or High across 5 function types (total of 8 different values)	2 SFP values allocated, one each to two function types
Range of FP values by category	Transactional functions are worth between 3 and 7 FP depending on type and complexity. Logical files are worth 7 to 15 FP depending on type and relative complexity	All transactional functions are considered to be EP and assigned 4.6 SFP. All data functions are considered to be logical files and assigned 7 SFP
Unit of measure	Function Points (FP)	Simple Function Points (SFP)
Convertibility	1 FP = 1 SFP	1 SFP = 1 FP