Software Solutions Symposium 2017

March 20–23, 2017

Methodology for the Cost Benefit Analysis of a Large Scale Multi-phasic Software Enterprise Migration

Bryce Meyer

Jerry Jackson

Jim Wessel

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Overview

- **Problem Set**
- **Cost Methods**
    - Decomposition vs Top Level
    - Comparative Methods
    - Parametric Methods
- **Adaptation of parametric method**
- **Accuracy**

Problem Set

- A large scale software development and utilization enterprise with over 800 software programs(each with distinct standards) that include enterprise business and data utilization, handheld, mobile, embedded safety critical and even instrument/sensor software items.
- Software programs are at every stage of the lifecycle from initial R&D and rapid prototyping through end of life decommissioning.
- Leadership has seen the savings enjoyed by other commercial entities who have migrated to a common software infrastructure with well defined interfaces and services. Potential savings in testing costs, design and coding costs etc. achieved through common core services that provide interoperability between programs needs to be determined but time is short and bottom up analysis of this many programs would be cost and resource restricted.
- ***How can leadership gain insight to whether the investment to build a common software infrastructure for its System of Systems environment will provide appropriate ROI and when will the Cost/Benefit finally invert?***

Traditional Project Estimation

System projects are often estimated:

1) Break down the project into very detailed fixed efforts
2) Cost each effort
3) Sum the costs
4) Adjust fire to meet funds available…i.e. delete efforts or limit efforts.

Software projects, especially those that use cyclic processes such as Agile, or are the sum of diverse services and applications:

1) Break down the project into very detailed fixed elements (i.e. modules, services, applications)
2) Cost each element
3) Sum the costs for each release in the cycle. Assume some level of reuse.

System of System software efforts are a hybrid of both project and software types, with elements over time.

Software Cost Models: Basic Types

Software targeted cost models fall into 4 basic types, as in the matrix below:

Approach / **Estimation Method**	**Decomposition:** Break down Software/ System/ SoS into constituent elements or project elements, small enough to estimate costs	**Systematic:** Estimate at a high level by system or project.
Comparative: Estimate by interpolation or extrapolation from historically similar efforts	Comparative estimation of each element in a Decomposition	Comparative estimation at the System/Project level
Parametric: Estimate with a mathematical model that incorporates factors from the effort	Parametric estimation of each element in a Decomposition	Parametric estimation at the System/Project level

Software Cost Models: Required Data

For each type below, different data is required ideally to complete the estimation:

Approach / Estimation Method	**Decomposition:** Break down Software/ System/ SoS into constituent elements or project elements, small enough to estimate costs	**Systematic:** Estimate at a high level by system or project.
Comparative: Estimate by interpolation or extrapolation from historically similar efforts	1) Complete breakdown at a low enough level for comparison 2) Comprehensive set of costs of comparative elements and data for correct comparison	Comprehensive set of costs of comparative systems/projects and data for correct comparison
Parametric: Estimate with a mathematical model that incorporates factors from the effort	1) Complete breakdown at a low enough level for comparison 2) Factors to feed the parametric model for each element.	Factors to feed the parametric model for the system/project

Software Cost Models: Required Data

Data requirements drive the selection of both approach and method, and often a hybrid of these methods is needed.

Software based projects early in development, with some risk in technologies, often lack data required for a traditional estimation, or a comparative/decomposition approach.

Systems of Systems that will use linked complex software systems. Savings from merging into a SoS are from eliminating duplication.

Data when available is often in ranges at best.

Notional System of System Merger

Leadership has decided, due to mission need and funding, to link a series of existing systems into a SoS using:

- Standards, including interfaces
- Cross-cutting decisions that dictate key engineering methods of the SoS
- Common Infrastructure (embodied as Computing Environments) that accepts functions or components of the existing systems as:
  - Services that become part of the infrastructure
  - Applications that use services from the infrastructure, or other applications.
- Remaining elements of the system are phased out/eliminated
- Software Components from existing systems are either applications or services and:
  - Wrapped into the services or applications in the new computing environment
  - Recoded into the services or applications in the new computing environment
  - Eliminated as duplicate
- Each Phase incorporates more standardization, more software migrated into the Infrastructure, and more capabilities migrated from systems

VERY LITTLE Detailed ESLOC data is Available: Most systems are still architecting the merger!

Estimation Method for SoS Software Development

Problem: What are the relative costs for various courses of action for the migration? Assume hardware and licensing are separate costs.

Given the scarcity of data for other methods, a hybrid approach is used to cost the example:

- Decomposition into the migrating systems and the computing environment infrastructure, by phase.
- Parametric estimation of each system (or example system) that will be migrated into the environment
- Parametric estimation of the computing environment infrastructure.

For this example, Expert COSYSMO was chosen as engine, wrapped in an Excel workbook.

Expert (and Academic) COSYSMO Inputs Required

How do we collect data to fill in all these variables?

Numeric Inputs	Easy	Nominal	Difficult
# of System Requirements	##	##	##
# of System Interfaces	##	##	##
# of Algorithms	##	##	##
# of Operational Scenarios	##	##	##

Qualitative Inputs	Scale (*=risky side)
Requirements Understanding	*Scale very low to very high
Architecture Understanding	*Scale very low to very high
Level of Service Requirements	Scale very low to very high*
Migration Complexity	*Extra High to Nominal
Technology Risk	*Extra High to Very Low
Documentation	Scale very low to very high*
# and Diversity of Installations/Platforms	*Extra High to Nominal
# of Recursive Levels in the Design	Scale very low to very high*
Stakeholder Team Cohesion	*Scale very low to very high
Personnel/Team Capability	*Scale very low to very high
Personnel Experience/Continuity	*Scale very low to very high
Process Capability	*Very Low to Extra High
Multisite Coordination	*Very Low to Extra High
Tool Support	*Scale very low to very high

Maintenance Off On	Off or On

Analysis and Verification prior to model run required

Gauge of E/N/D is critical for reliable results.

Caution Required- Some variables very sensitive and greatly alter results

5 Scales in the Qual. inputs: Can be confusing to many users…

Obtaining COSYSMO Qualitative Input Variables

A survey can collect the data required for a range of possible costs for each of system +infrastructure:

- Derive a series of questions (~77) for each phase for the system/environment that can map to the COSYSMO Qualitative inputs.
  - A common scale of Very High Risk to Very Low Risk is used for every question.
  - Questions derived from SEI best practices linked to Qualitative input area.
  - Multiple survey questions to each COSYSMO input compensate for interpretation.
  - Costing Team helps each respondent with questions to assure understanding.
  - A series of mathematical maps convert the risk responses to Qualitative Inputs.

Obtaining COSYSMO Numeric Input Variables

Two options are available for filling in the four Numeric Inputs:

1) Assess the count of Easy, Nominal, and Difficult for each Input for each phase: This requires an intimate understanding of the future phases.

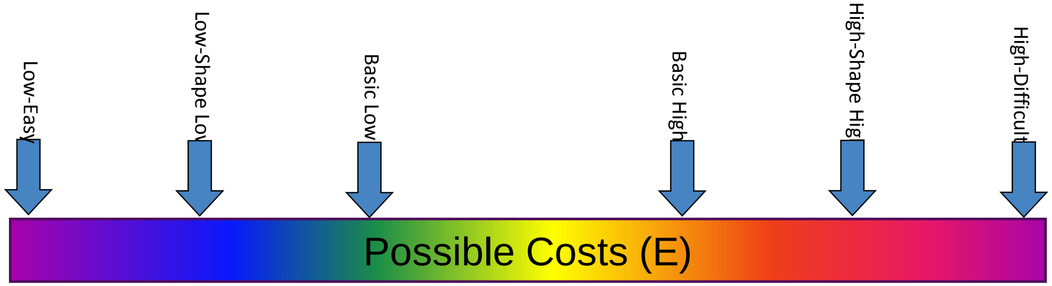
2) Assess a high and low range for each input per phase, then assess a low end (all easy), high end (all difficult), and normally distributed estimate tilted difficult (i.e. 4% Easy, 18% Difficult, 79% Nominal).

After initial interviews with each system, an option can be chosen. Due to data availability, Option #2 is assumed selected.

Numeric Inputs	Easy	Nominal	Difficult
# of System Requirements	##	##	##
# of System Interfaces	##	##	##
# of Algorithms	##	##	##
# of Operational Scenarios	##	##	##

Cost Estimates in the Spread (for each phase interval)

Estimate (E)	Quantitative Value used (U_Q)	Quantitative Difficulty Used
Low-Easy	Low (L_Q)	All Easy
Low-Shape Low	Low (L_Q)	Allocated by 1-sigma Easy tilted Normal Curve (set to 18% E, 79% N, 4% D)
Basic Low	Low (L_Q)	All Nominal
Basic High	High (H_Q)	All Nominal
High Shape High	High (H_Q)	Allocated by 1-sigma Difficult tilted Normal Curve (set to 18% D, 79% N, 4% E)
High-Difficult	High (H_Q)	All Difficult



Filling in the Surveys

Given timing, a sample of systems for each computing environment can be used for quick analysis, then other systems assessed as similar to one of the surveyed systems.

Survey Questions: Strong and clear definitions of each question vs. risk level, with examples and detailed long definitions will be required as common rulesets.

Numerical Values: Pick common rules given the comparative nature (chose between alternatives):

- Interfaces: a 'Rosetta Stone' is also required to assess interfaces. Rules set for what defines an interface for the purposes of the study. Interface context used: in the migrated sense, what interfaces enter or exit the service or apps made from the system?
- Requirements: What level of requirements for the system (remaining after projected migration) will be used for the estimates?
- Algorithms: What core functions will be migrated, i.e. how many functions of the system can be said to perform a data conversion or calculation.
- Operational Scenarios: What level of scenarios, remaining after migration, will migrated system elements perform? The level set should be common to operational test organization definitions.

Qualitative Survey

##	Qualitative Variable	Associated Questions	What is the risk with this item? (Very Low to Very High) For v2 to v3	What i… Ve…
1	Requirements Understanding	Are the developers, integrators, and architects (at all levels) familiar with the implications of the Infrastructure and Environment requirements and related guidance with respect to their effect on the Environment (or perhaps System level) build? These requirements may originate from external or internal stakeholders.	nominal / Avg	nomin
2	Requirements Understanding	Are the developers, integrators, and architects (at all levels) familiar with the implications of each existing system's capabilities on the Environment software, for each Environment version?		nomin
3	Requirements Understanding	Are the developers, integrators, and architecture (at all levels) familiar with the implications on Environment internal interfaces and Control Point (Environment to Environment) interfaces?		nomin
4	Requirements Understanding	Are the developers, integrators, and architects (at all levels) familiar with the security implications for the Environment (assuming the PM or capability developer is required to get an ATO or CON for their specific system or capability)?	nominal / Avg	nomin
5	Requirements Understanding	Are the developers, integrators, and architects (at all levels) familiar with implications on network, software, and hardware performance at the system level from the requirements imposed from the Infrastructure and Environment level, for each version?	nominal / Avg	nomin

VH risk=no familiarity, H=limited familiarity, N=somewhat familiar, Low=mostly familiar, VL risk=fully familiar

Adding a Test Factor

Preliminary Analysis of COSYSMO revealed a likely under-estimate of Systems of Systems operational testing for:

- Linkages between systems
- Network effects on software testing
- Threads that cross multiple systems

As a result, we need a 'test factor', i.e. a multiplier against each system or environments cost estimate.

Our work estimated this factor at 10%-24%, depending on mission threads using the system or environment infrastructure post-migration.

Setting a Floor and Ceiling

It is important to understand that even given survey responses (especially given the fact that people in systems may not want migration), there will be limits for a system and environment, to prevent a system from unfairly bias the costs estimates:

1) Cross-Cutting Concerns assessed for each computing environment must be met for the overall infrastructure. These are a floor estimate for interfaces.

2) Each Environment must also provide personnel at minimal levels to architect, engineer, and project manage the environment, and work with other environments to embody the infrastructure. This minimum sets a minimum cost for the environment.

3) Excessive numerical input numbers (provided to manipulate the costs) should be leveled to a ceiling set ahead of time using a second set of impartial engineers familiar with the systems, and with artifacts as evidence. Given this ceiling, choices will have to made to determine if the excess is actually a realistic estimate, or a flippant one.

Summing the Results

Each Course of Action has a set of Environments and Infrastructure choices, with assigned systems.

For each course of action, a high and low risk response for each phase, should be estimated:

- Environment Infrastructure is estimated at high and low, set within floor and ceiling levels.
- Sample systems estimated at high and low given floor and ceiling.
- Remaining systems to be migrated into the environment are assigned to a sample system as a multiplier.
- Total of Environment Infrastructure plus multipliers for each sample system estimate are totaled in high and low estimates for each phase.
- Multiply by (1+Test Factor).
- Total phases to get a spread of relative cost estimates over time.

Using the Results

For each course of action, a range of cost estimates by phase are not the only product.

Findings can emerge from the process of estimation:

- An assessment of the level of communication and understanding from top level engineering to system level engineers.
- Rough estimate of readiness of software in systems for migration. Many systems may have lost documentation and requirements

Systems responding will essentially self-assess, and may implement quality improvements as a result.

System respondents will be forced to begin a ‘keep or pitch’ analysis.

Estimates are relative costs in most cases, that can be compared to decomposition/comparative methods to tighten the range of estimates.

Next Steps

Relative costs can be tightened to improve fidelity by:

Digging deeper into the migration plan, including better estimates of ‘keep and pitch’ may allow ESLOC estimates, permitting use of COCOMO, QSM, PRICE or other tighter parametric methods, or historical/comparative methods.

Making more realistic choices for interfaces and cross cutting concern costs: How many will be crafted by one environment and provided to other environments

Using deployment to estimate hardware costs and licensing costs.

BACKUP SLIDES

Expert (and Academic) COSYSMO Inputs Required

Numeric Inputs	Easy	Nominal	Difficult
# of System Requirements	##	##	##
# of System Interfaces	##	##	##
# of Algorithms	##	##	##
# of Operational Scenarios	##	##	##

Qualitative Inputs	Scale (*=risky side)
Requirements Understanding	*Scale very low to very high
Architecture Understanding	*Scale very low to very high
Level of Service Requirements	Scale very low to very high*
Migration Complexity	*Extra High to Nominal
Technology Risk	*Extra High to Very Low
Documentation	Scale very low to very high*
# and Diversity of Installations/Platforms	*Extra High to Nominal
# of Recursive Levels in the Design	Scale very low to very high*
Stakeholder Team Cohesion	*Scale very low to very high
Personnel/Team Capability	*Scale very low to very high
Personnel Experience/Continuity	*Scale very low to very high
Process Capability	*Very Low to Extra High
Multisite Coordination	*Very Low to Extra High
Tool Support	*Scale very low to very high

Maintenance Off On	Off or On

Analysis and Verification prior to model run required

Gauge of E/N/D is critical for reliable results.

Caution Required- Some variables very sensitive and greatly alter results

Affects of Qualitative Variables on COSYSMO

(swing from baseline between best and worst, for the overall model)