Developing a Cost Estimation Probability Model of a Large Multi-Year System – An Experience Report

NDIA 2012

Kathleen Dangle, ***Madeline Diep***, and Forrest Shull
Fraunhofer Center for Experimental Software Engineering

Fraunhofer USA
Center for Experimental Software Engineering

Context of Our Experience

- Program characteristics:
  - Aerospace domain
  - Large and mission-critical system
  - Prime contractor with many partners and subcontractors
  - Distributed development
  - Government oversight
  - Software-intensive and COTS heavy
- Cost estimation in preparation for Key Decision Point (KDP) to enter *Concept & Technology Development*

Why A Probability Distribution Model

- Cost estimation is an important part of program planning and tracking
- But it can be very difficult to do accurately
  - Single-point estimation does not account for uncertainties in the estimation sources and errors in the model
- Probability distribution model helps to understand the likelihood of achieving the point estimate
  - Required to budget the cost estimate at the 70% confidence level.

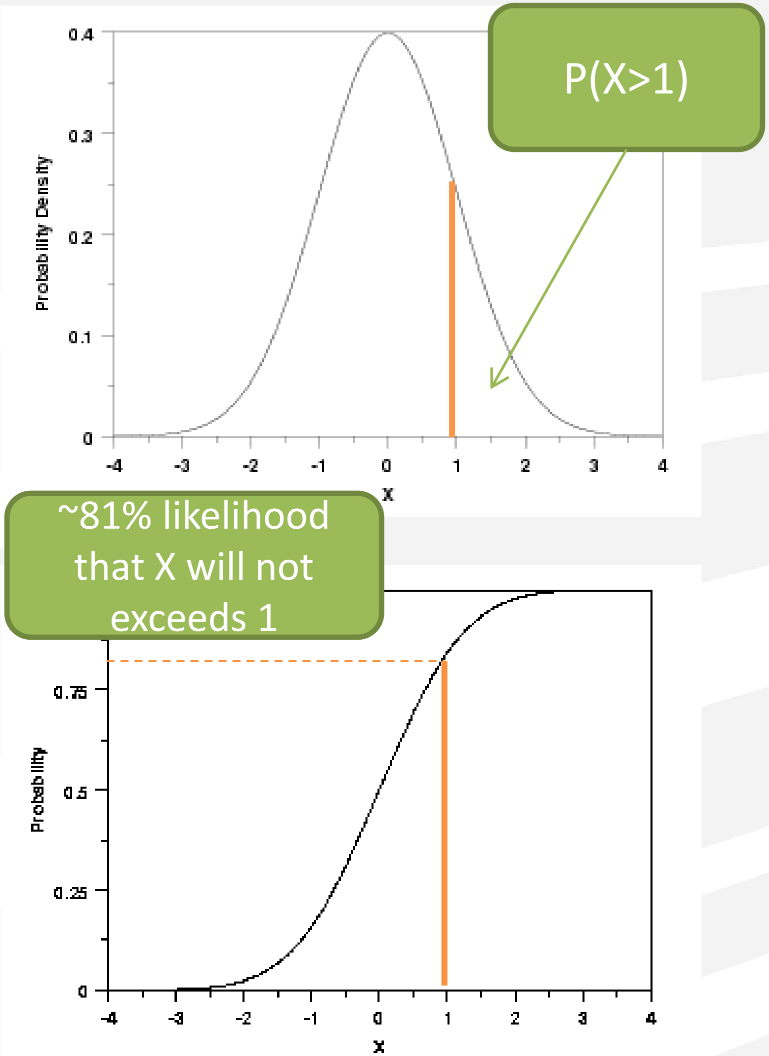
GAO Guidelines on Cost Estimation

Include probability distributions...As outlined in the GAO Publication*

1. Determine the program cost drivers and associated risks;
2. Develop probability distributions to model various types of uncertainty (for example, program, technical, external, organizational, program management including cost estimating and scheduling);
3. Account for correlation between cost elements to properly capture risk;
4. Perform the uncertainty analysis using a Monte Carlo simulation model;
5. Identify the probability level associated with the point estimate;
6. Recommend sufficient contingency reserves to achieve levels of confidence acceptable to the organization; and
7. Allocate, phase, and convert a risk-adjusted cost estimate to then-year dollars and identify high-risk elements to help in risk mitigation efforts.

*GAO Cost Estimating and Assessment Guide, GAO-09-3SP, March 2009

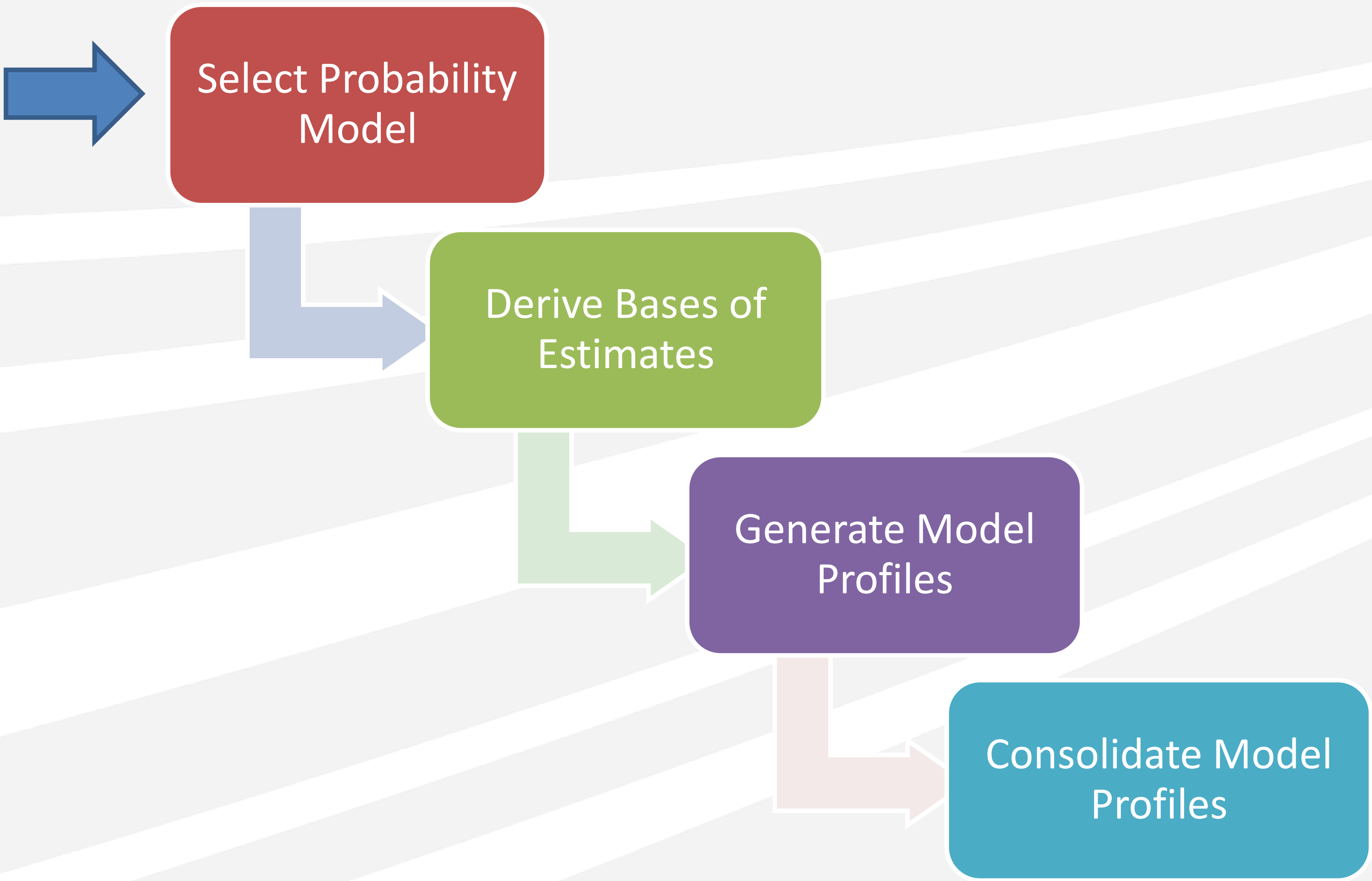
About Probability Distribution Models



- Usually represented in two views:
  - Probability Density Function (PDF)
  - Cumulative Density Function (CDF or S-curve)

- Describe the probability that a variable have a value less than or equal to *x*

Building the Model



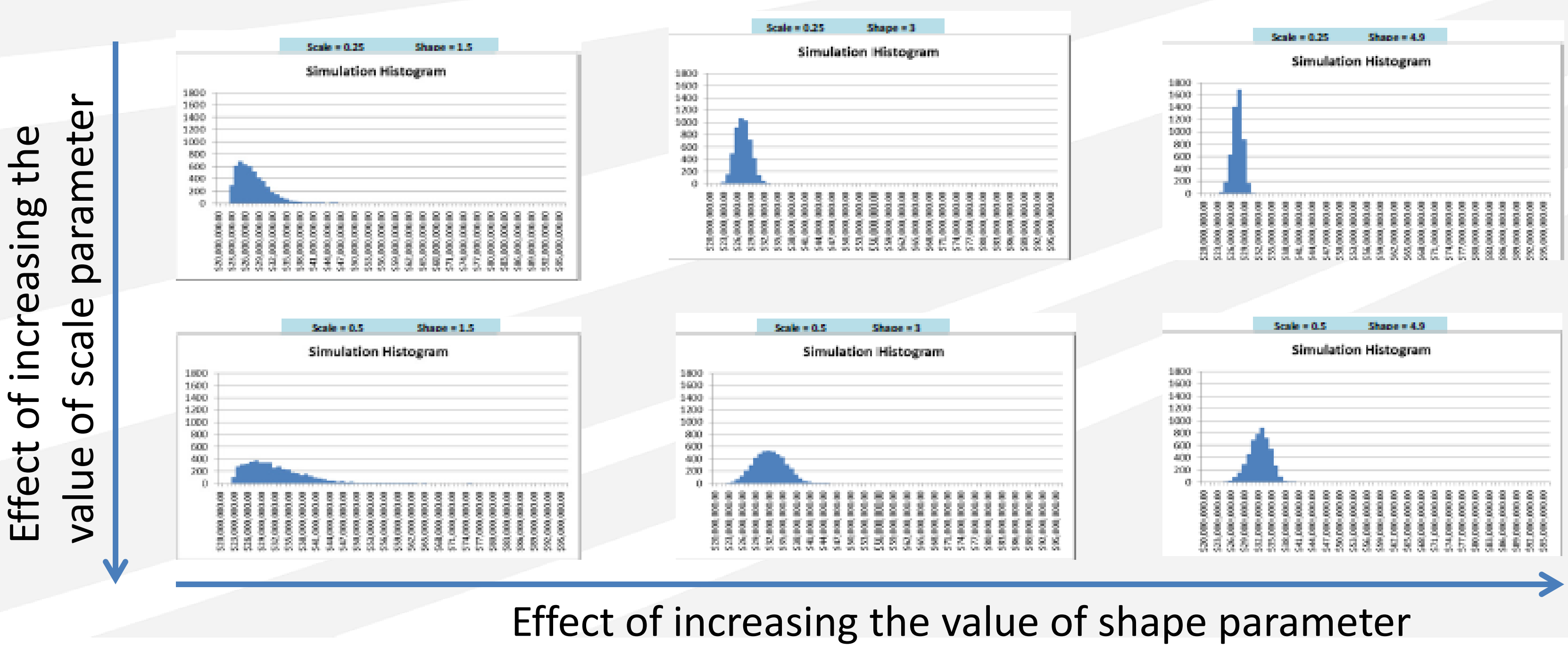
Step 1: Select Probability Model

- A number of probability distribution models are available to represent cost risks for the different cost elements.

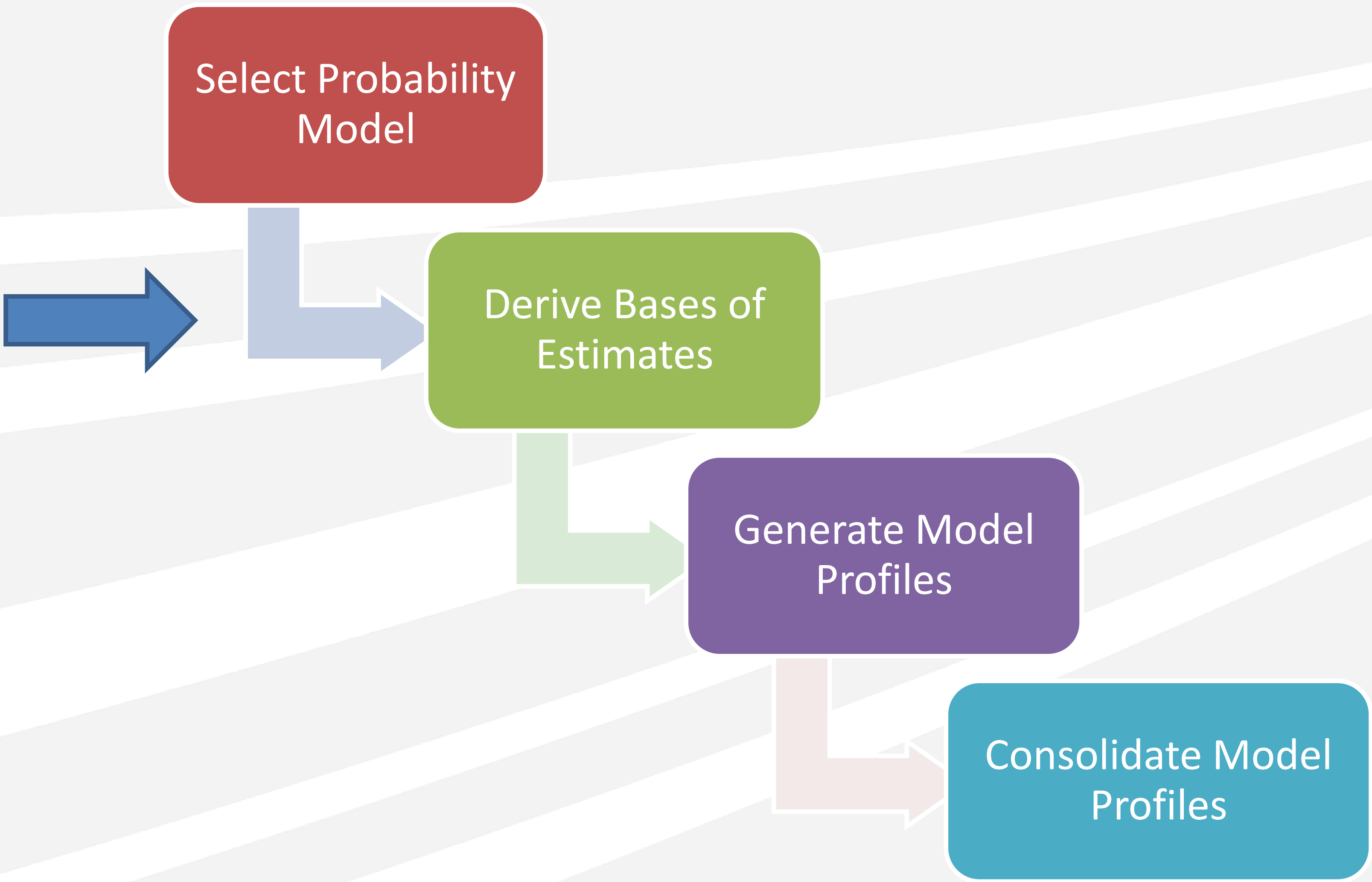
- We selected the **Weibull probability** distribution
  - Traditionally used for reliability models
  - Flexible -- its three parameters can be adjusted to represent distribution curves such as the normal, logarithmic, Rayleigh, and exponential
  - A single model with the ability to accommodate multiple cost model profiles

On the Weibull Distribution

- Three parameters of Weibull model:
  - **Shape**: Affects the shape of the model, as well as slope of the model
  - **Scale**: Increasing its value while holding “Shape” variable constant has the effect of stretching out the probability model
  - **Location**: Provides an offset for the starting value of the (cost) variable



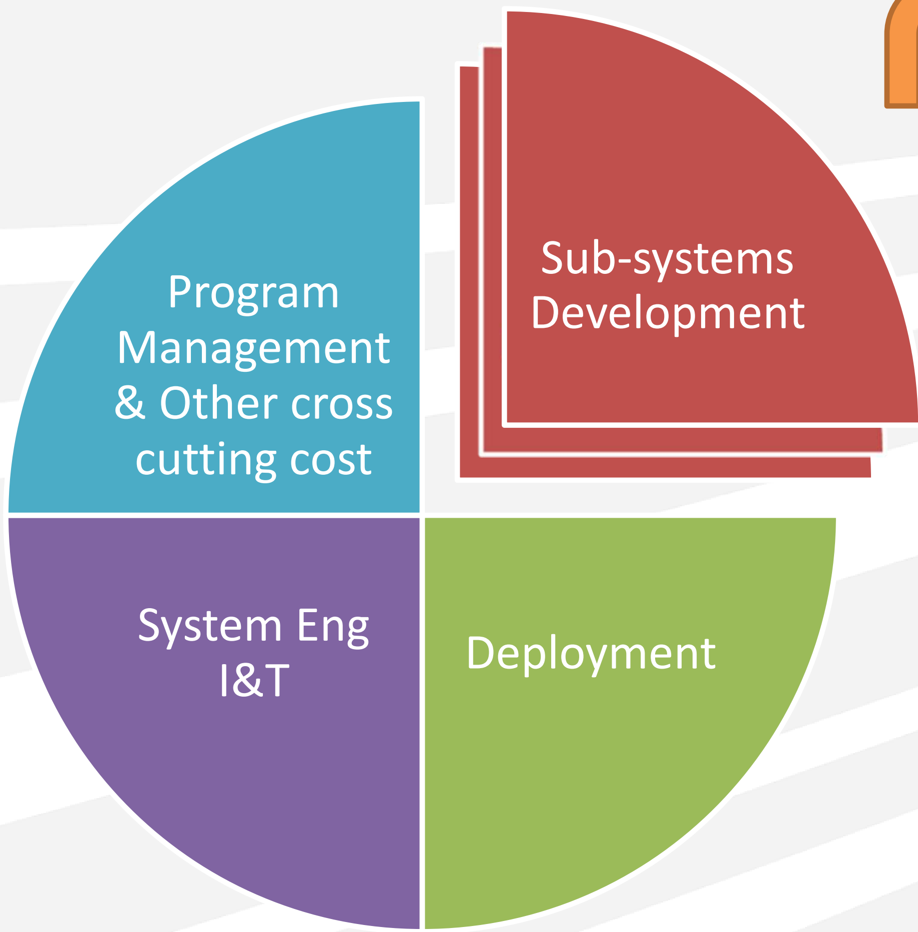
Building the Model

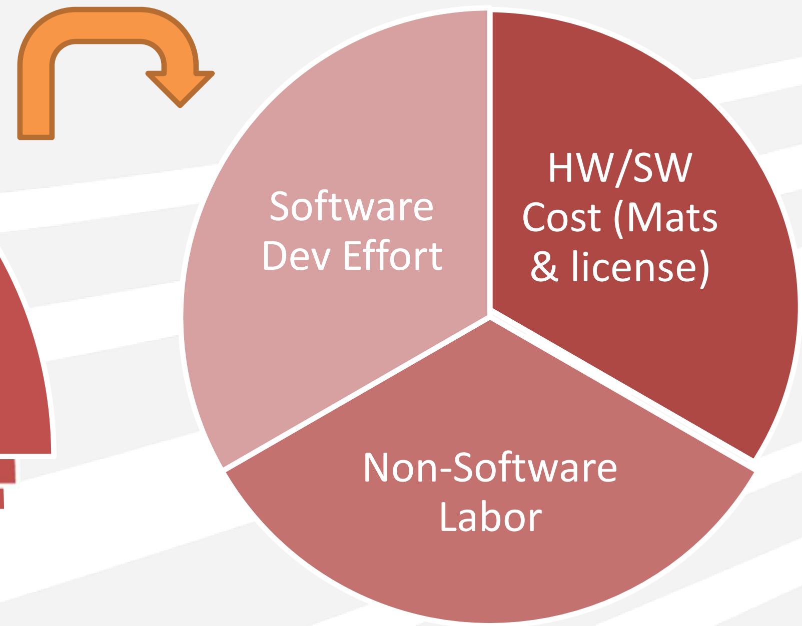


Step 2: Derive Bases of Estimates

- Large Program = Large and deeply nested WBS
  - Impractical to create a cost model for each item in the Work Break-down Structure (WBS)
- Created "buckets" for the WBS items and generated a cost model for each bucket
  - Items in a bucket should, intuitively, have similar types of uncertainty or risks
  - Buckets should be general enough to contain all items under Level 3 or 4 WBS elements

Buckets of Cost Elements

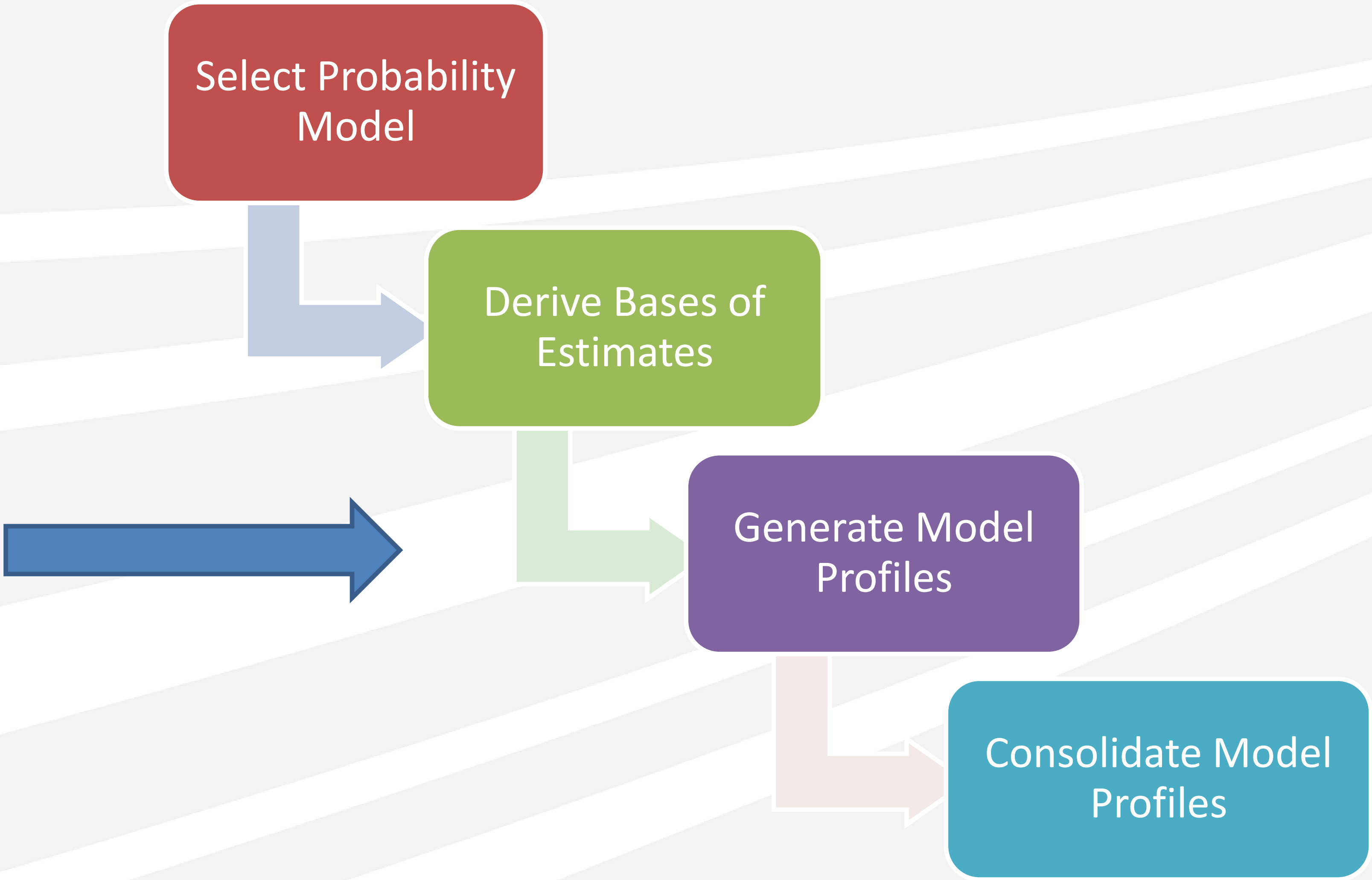




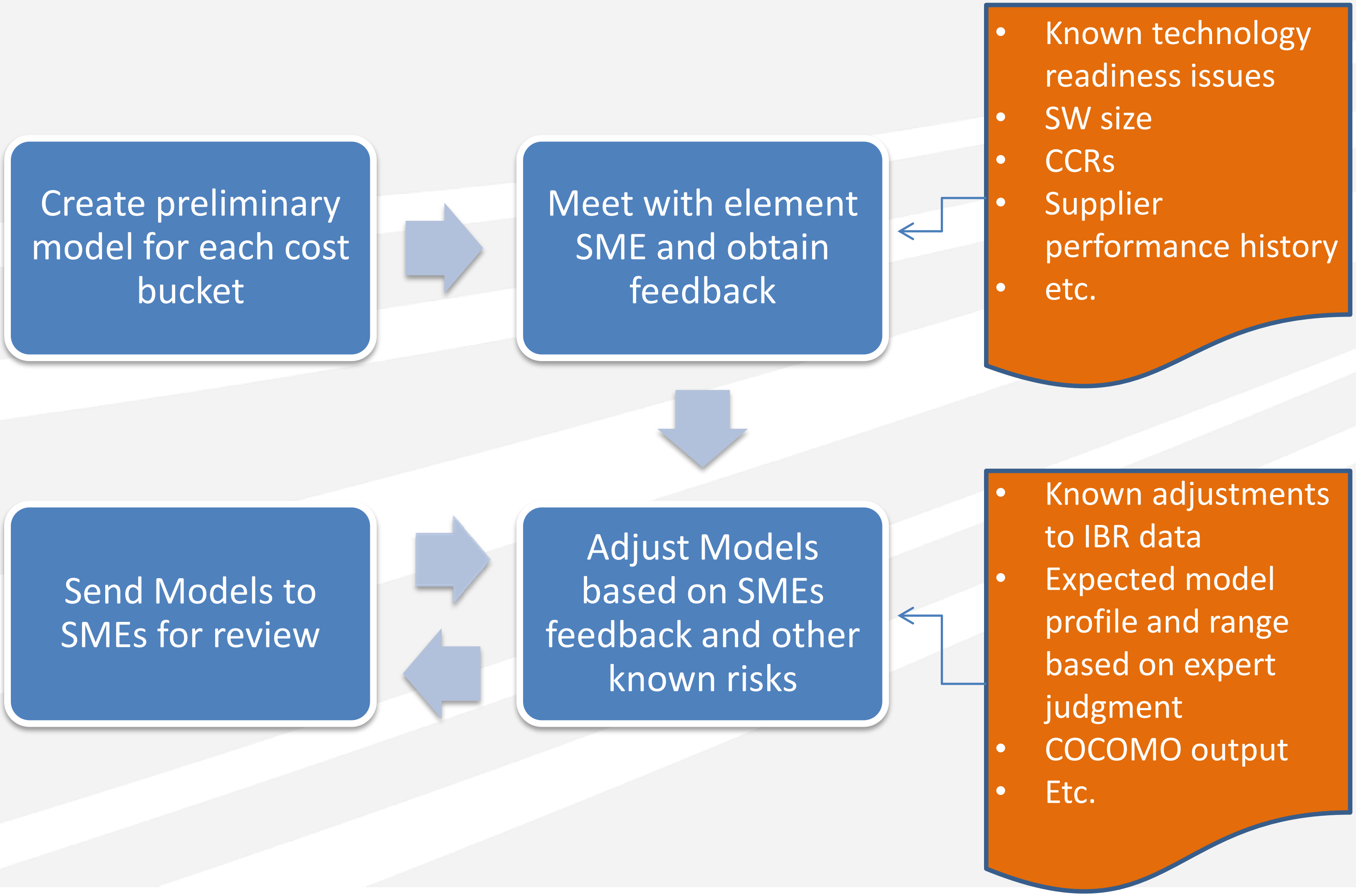
SW development effort includes COTS labor, VHDL development effort, maintenance, support (documentation)

Non-SW Labor includes sub-system engineering labor, I&T, COTS and HW operational related labor (handling, shipping)

Building the Model



Step 3: Generate Model Profiles



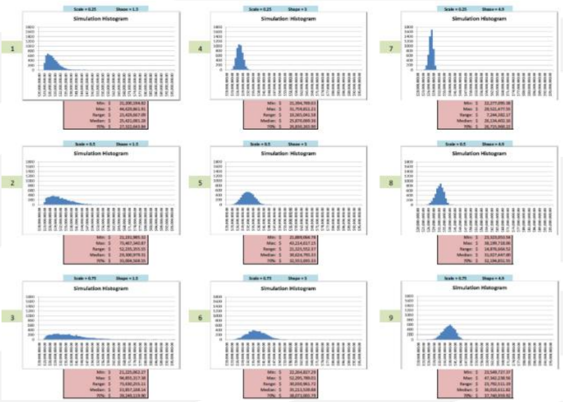
Preliminary Model

- For all “buckets”, set the ***location*** variable to be the original Initial Baseline Review (IBR) estimation
- Criteria for setting the ***scale*** and ***shape*** parameters:
  - For sub-systems’ software cost bucket
    - Software labor cost tends to have larger uncertainty and to be prone to underestimations → the model is skewed to the left (median value >> IBR estimates)
    - Used variability in SW sizes (likely and high estimates) to adjust the skewness → the higher the variability, the more skewed the model
  - For sub-systems’ hardware/software cost bucket
    - HW/SW material and licensing cost is generally stable where variability comes from amount of equipment to acquire → the model was skewed to the right (median value > IBR estimates)
  - For sub-systems’ non-SW labor & program management (cross cutting concerns), I&T, and Deployment bucket
    - Some variability is expected → used normal distribution model

Fraunhofer USA
Center for Experimental Software Engineering

Getting Subject Matter Expert Input

- Talked to 1-2 SMEs from each major sub-components
- Presented the preliminary model
- To facilitate the discussion, provided reference model of a set of Weibull distribution profiles



- Captured feedback:
  - The range and skewness of the model
  - Justification for the profile:
    - Known issues, other estimation analyses, perceived risks

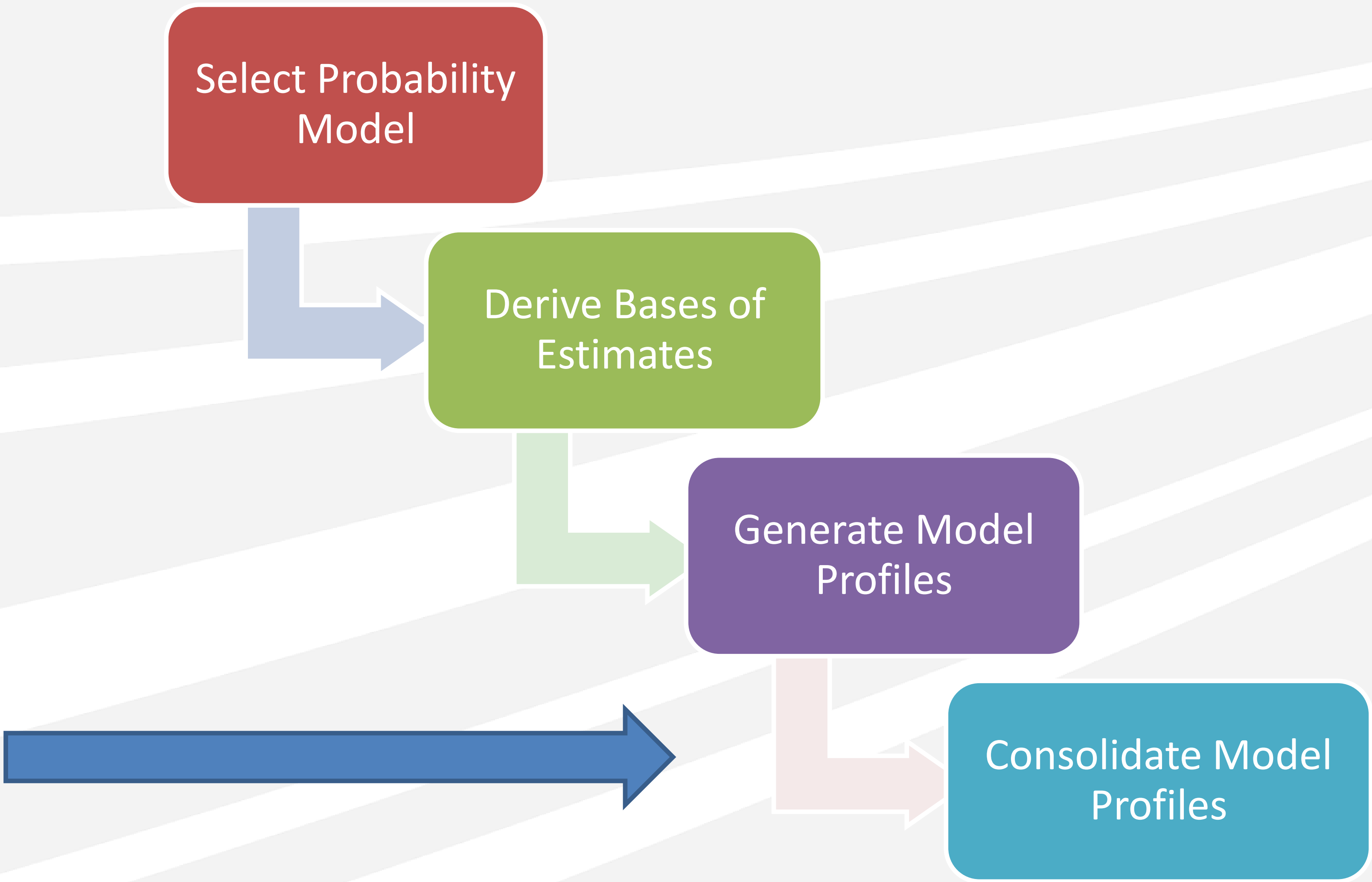
Adjusting the Model

- Incorporated feedback from the SMEs and any known estimate adjustments thus far, as well as the dollar value of risk exposure

- Performed sanity check:
  - Use COCOMO analysis to estimate software development effort/cost based on *estimated* software size
  - Ensure that the model "includes" the COCOMO output

Fraunhofer USA
Center for Experimental Software Engineering

Building the Model



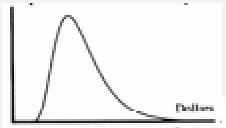
Step 4: Consolidate Model Profiles

Generate a cost model for each sub-system by "summing" SW effort, HW/SW cost, and non-SW labor

Correlate cost-models of the sub-systems

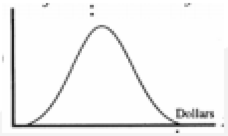
Generate the Program cost model by "summing" all the cost-models

SW Effort Model



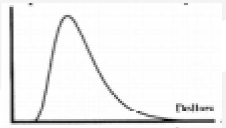
+

HW/SW Cost Model

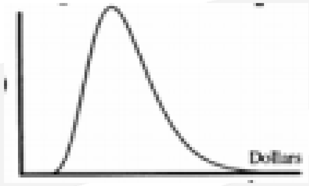


+

Non-SW Model

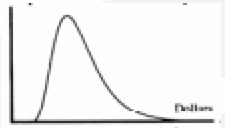


=



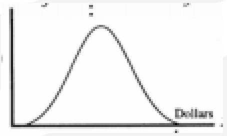
A sub-system cost model

Sub-system 1 Model



+

Sub-system 2 Model

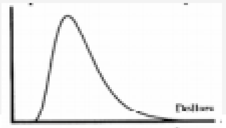


+

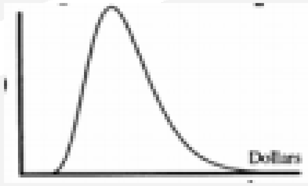
…

+

Program Management Cost



=



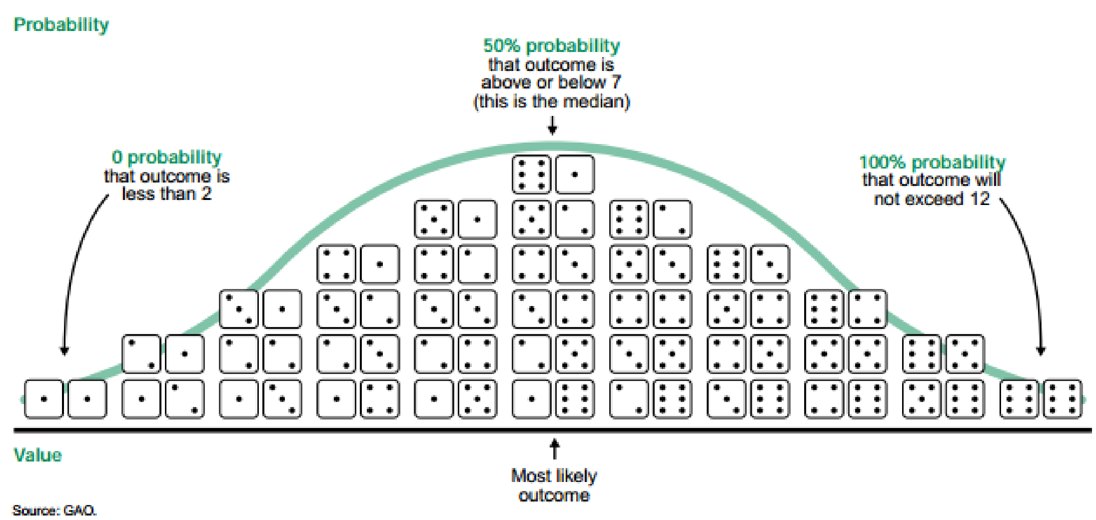
The Program cost model

Fraunhofer USA

Center for Experimental Software Engineering

"Summing" the Models

- Used *Monte Carlo* simulations to "sum" the models
  - Repeatedly generated a random value following the defined distribution model
  - For each sampling, summed the generated value across all the distribution models
  - Created histograms of the sum generated from all the samplings
- Note: The number of iterations/samples can significantly affect the range of the model
  - We used 10,000 iterations/samples, and still observed some sensitivity in the produced models



Fraunhofer USA
Center for Experimental Software Engineering

Parting Thoughts

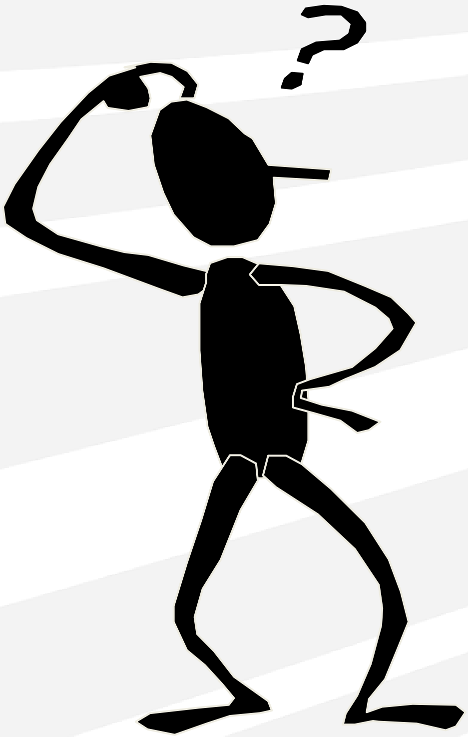
- We described our approach for generating cost-estimation probability distribution models
- Some lessons learned:
  - A WBS that makes key cost drivers “visible” makes it easier to “bucket” cost items in a manageable way.
  - Weibull is a flexible model, but may not be as popular as other models → lack of accessible statistical tool support for performing more sophisticated activities, such as model correlation.
- Program is still ongoing
  - The cost model have been vetted by a separate cost policy team
  - *We are collecting the data now for the quantitative validation*

List of Abbreviations

- CCR: Contractor Cost Report
- I&T: integration and Test
- IBR: Initial Baseline Review
- SME: Subject Matter Expertise
- VHDL: VHSIC (Very High Speed Integrated Circuit) Hardware Description Language

Fraunhofer USA
Center for Experimental Software Engineering

Questions?



Contact Information

Madeline Diep

mdiep@fc-md.umd.edu

240-487-2937

Fraunhofer USA
Center for Experimental Software Engineering

Backup

Fraunhofer Center Maryland

- A not-for-profit ***applied research & technology transfer*** organization
- Mission: ***Advance real-world software practices*** via empirically validated research into software-engineering technologies and processes

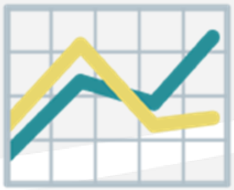
- Work closely with the customer to develop unique, ***innovative solutions*** within their business context
- Purveyor of ***best practices*** to organizations inside and outside of the software industry
- Affiliated with Fraunhofer-Gesellschaft in Germany and University of Maryland at College Park

Fraunhofer USA
Center for Experimental Software Engineering

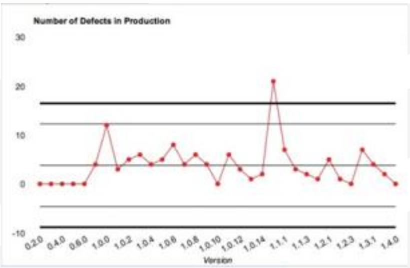
University of Maryland 1856

Fraunhofer Center Maryland

- **Competences**
  - Security, reliability & safety analysis
  - Software design & architecture analysis
  - Software testing, verification & validation
  - Technology evaluation
  - Software product & process evaluation
  - CMMI training & coaching
  - Program, project, and risk management



GQM+Strategies®



- Deep **expertise in**
  - Goal-driven, measurement-based process and product improvement
  - Software defect causal analysis and resolution
  - Process improvement and best practices