Presented at the 2008 SCEA-ISPA Joint Annual Conference and Training Workshop - www.iceaaonline.com

estimate

estimate • analyze • plan • control

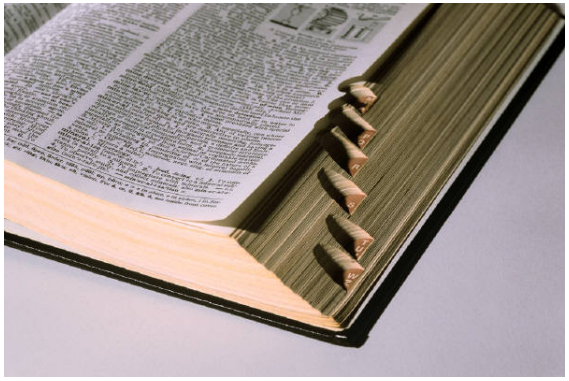
Software Total Ownership Cost: Development Is Only Part of the Equation

Daniel Galorath

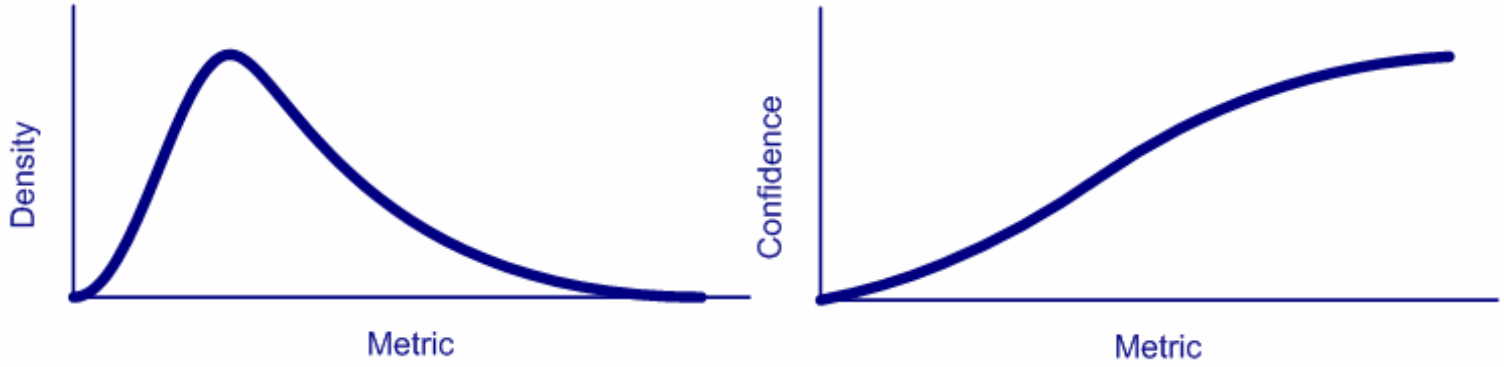
SEER® by GALORATH

An Estimate Defined

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

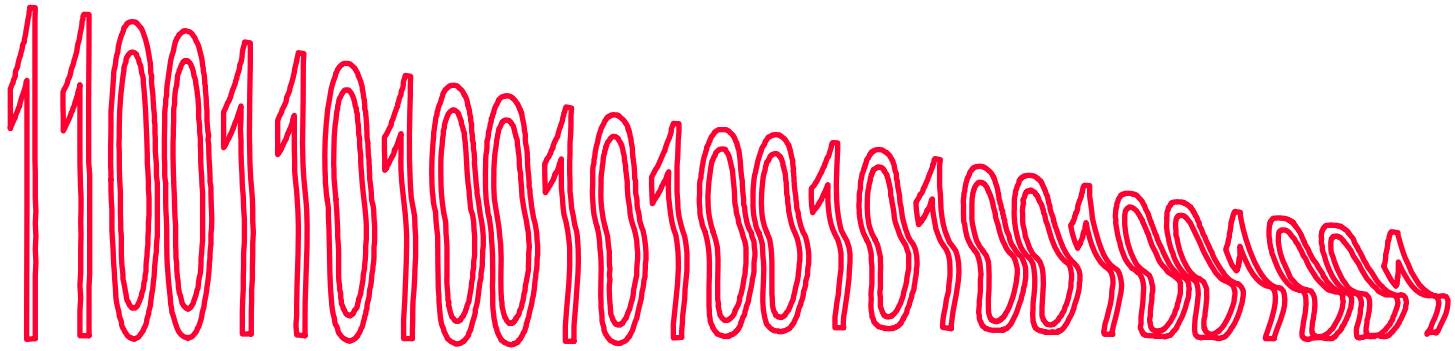


- *A well formed estimate is a distribution*
- *A well structured plan defines probability*



Maintenance Defined

- Dictionary: "The work of keeping something in proper order"
- Software maintenance is different from hardware maintenance because:
  - Software doesn't physically wear out, but...
  - Software often gets less useful with age and...
  - It may be delivered with undiscovered flaws
- Software maintenance is: "The process of modifying existing operational software while leaving its primary functions intact."



Poor Estimates Effects on Projects

SEER by GALORATH

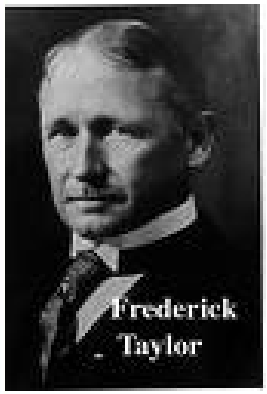
- Inaccurate estimates can reduce project success:
  - Poor implementations
  - Critical processes don't scale
  - Emergency staffing
  - Cost overruns caused by underestimating project needs
- Scope creep from lack of well defined objectives, requirements, & specifications
  - Forever changing project goals
  - Frustration
  - Customer dissatisfaction
  - Cost overruns and missed schedules
  - Project Failures
- Poor estimates & plans are root cause of program risk

However, the most important business decisions about a software project are made at the time of *minimum knowledge* and *maximum uncertainty*

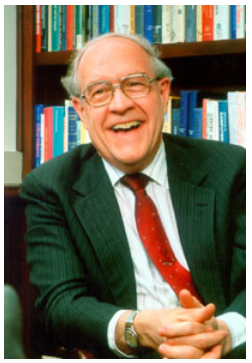
Some of Dan's Heroes Throughout Time

SEER by GALORATH

- Frederick Taylor: The Principals of Scientific Management 1901 "Let data and facts do the talking"
- W. Edwards Demming: "In God We Trust… All Others Bring Data"
- Frederick Brooks: "There is an incremental person when added to a software project that makes it take longer"
- Ed Yourdon: "Avoiding Death Marches in Software Projects"
- Steven Covey: "Sharpen the Saw" Focus on improvement



"In God we trust,
all others bring data."
- W. Edwards Deming

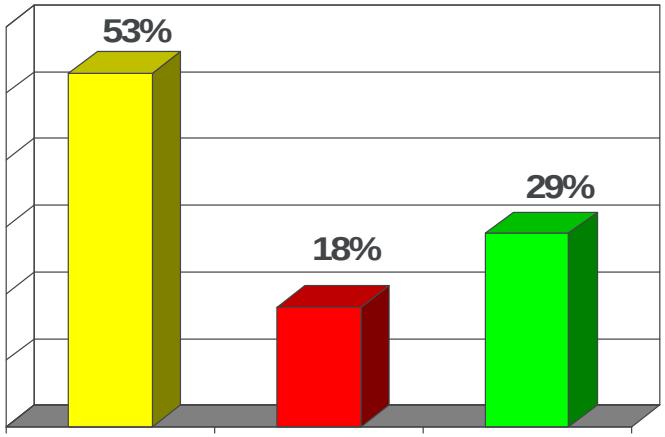


Top Management Directive "Run IT like a Business"

SEER by GALORATH

$255 billion spent on IT projects

- 53% challenged
- 18% failed
- 29% successful



Standish Group, Chaos Report, 2004 Third Quarter findings

2005 Cutter Consortium software project survey reported:

- 62% overran original schedule by more than 50%
- 64% more than 50% over budget;
- 70% had critical product quality defects after release

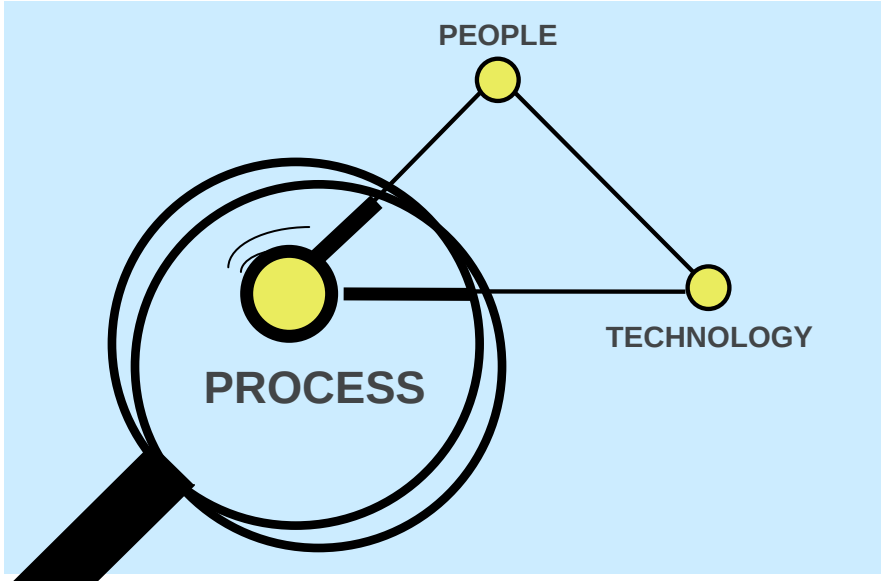
$55 billion of U.S. IT budgets wasted annually

Averages 22% of IT organizations budget

People, Process, Technology Are Keys

Source CMMI Tutorial

- Everyone realizes the importance of having a motivated, quality work force but...
- ...even our finest people can't perform at their best when the process is not understood or operating "at its best."



Major determinants of product cost, schedule, and quality

Estimation Role In CMMI

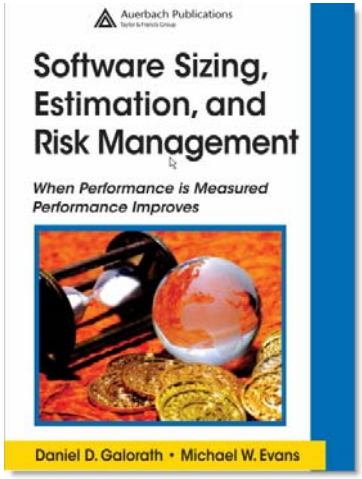
SEER by GALORATH

Quantitative Project Management

A Foundation of Risk Management

10 Step Software Estimation Process:
Consistent Processes = Reliable Estimates

SEER by GALORATH



1. Establish Estimate Scope
2. Establish Technical Baseline, Ground Rules, Assumptions
3. Collect Data
4. Estimate and Validate Software Size
5. Prepare Baseline Estimates
6. Review, Verify and Validate Estimate
7. Quantify Risks and Risk Analysis
8. Generate a Project Plan
9. Document Estimate and Lessons Learned
10. Track Project Throughout Development

Step One: Establish Estimate Scope and Purpose

- Define and document estimate expectations, scope & Purpose
  - Provides baseline against which to gauge future change effects
  - Reduces misunderstandings & contradictory assumptions
- Estimate should be considered a living document
  - As projects change, data changes or new information becomes available, it should be documented and factored into the estimate in order to maintain the project's integrity
  - This is not a copout…. Plans must be made from the estimate

Step Two: Establish Technical Baseline, Groundrules, & Assumptions

- Functionality included in the estimate or range must be established
  - If detailed functionality is not known, groundrules and assumptions state what is and isn't included in the estimate
  - Issues of COTS, reuse, and other assumptions should be documented as well
- Groundrules and assumptions form the foundation of the estimate
  - Although early at early stages they are preliminary and rife with uncertainty, they must be credible and documented
  - Review & redefine as the estimate moves forward

Step Three: Collect Data

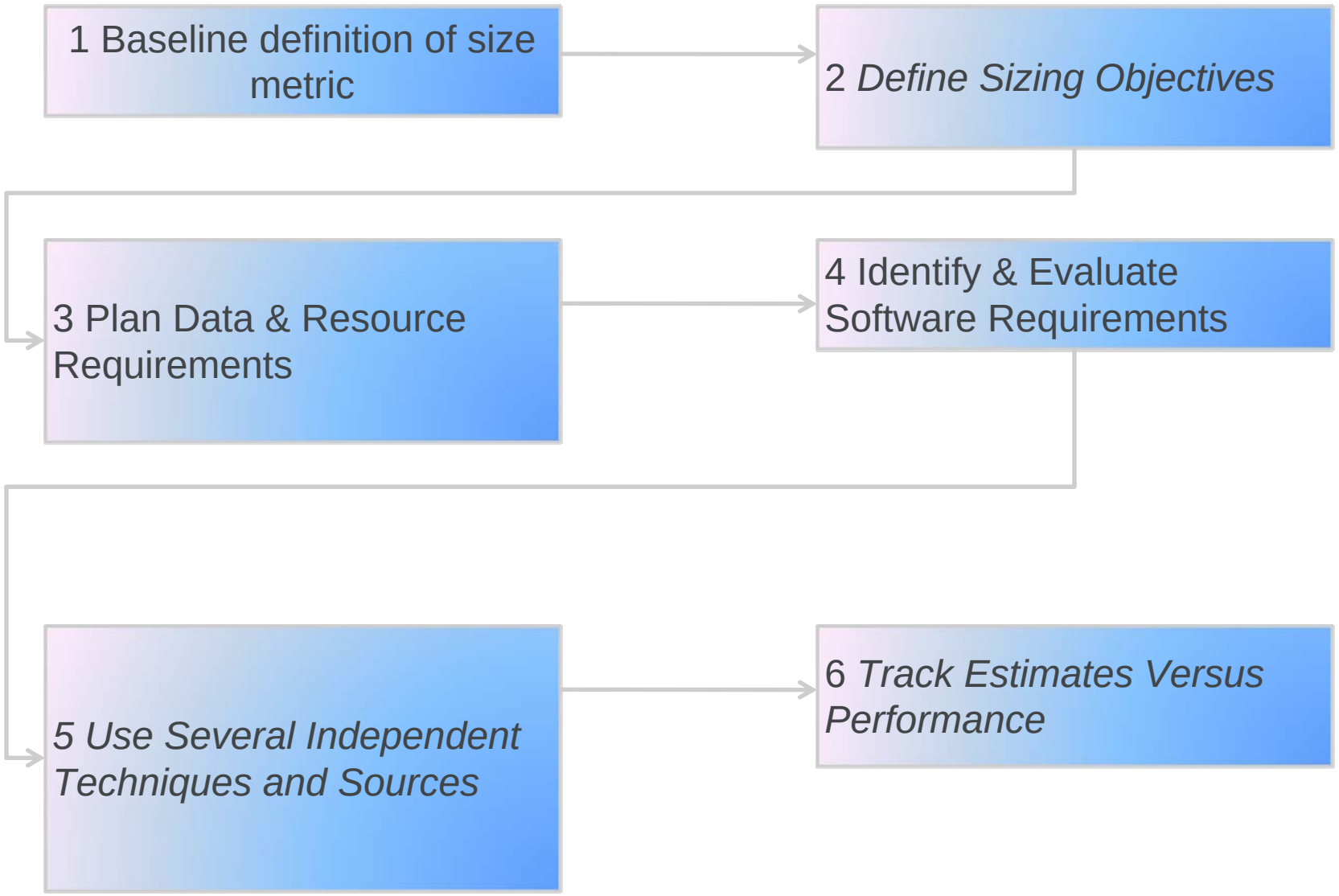
- Software Data Collection Process key considerations
  1. Motivate potential data providers to participate
  2. Avoid nondisclosure agreements containing clauses requiring exclusivity or destruction of data if you can
  3. Provide data collection forms and instructions beforehand, in both hard copy and electronic formats
  4. Provide clear definitions & recognize providers may not read them
  5. Identify which data are *required*, *highly desirable* or *desirable*
  6. During interview confirm data is realistic and valid
  7. Grade to indicate confidence
  8. Normalize data via well-documented process & keep the raw

Fundamental Metrics For Estimation, Planning & Control

SEER by GALORATH

- Size
  - AKA Volume, Mass
  - Units: Source Lines of Code (SLOC); Function Points (FP) Use Cases
  - New versus rework
  - COTS & Packages
- Effective Technology
  - AKA Productivity Potential, Efficiency
  - Units: none
- Time
  - AKA Duration, Schedule
  - Units: Calendar Months, Calendar Weeks
- Effort
  - AKA Work, Labor
  - Units: Staff Months, Staff Hours
- Cost
  - AKA Budget, Money
  - Units: $, other currencies
- Staffing
  - AKA Manpower Loading
  - Units: FTE People
- Defects
  - AKA Reliability, Quality
  - Units: Defect Count

Ideal Size Projection Takes Time
6 Step Process



Size Study Methodology

SEER by GALORATH

Evaluate All Sources of Software Size…

New Size

Pre-existing Size (rework)

Generated Code

COTS/GOTS

Glue Code

Integrated Code

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

Expert Judgment

Functional Analysis

SEER-AccuScope Analysis

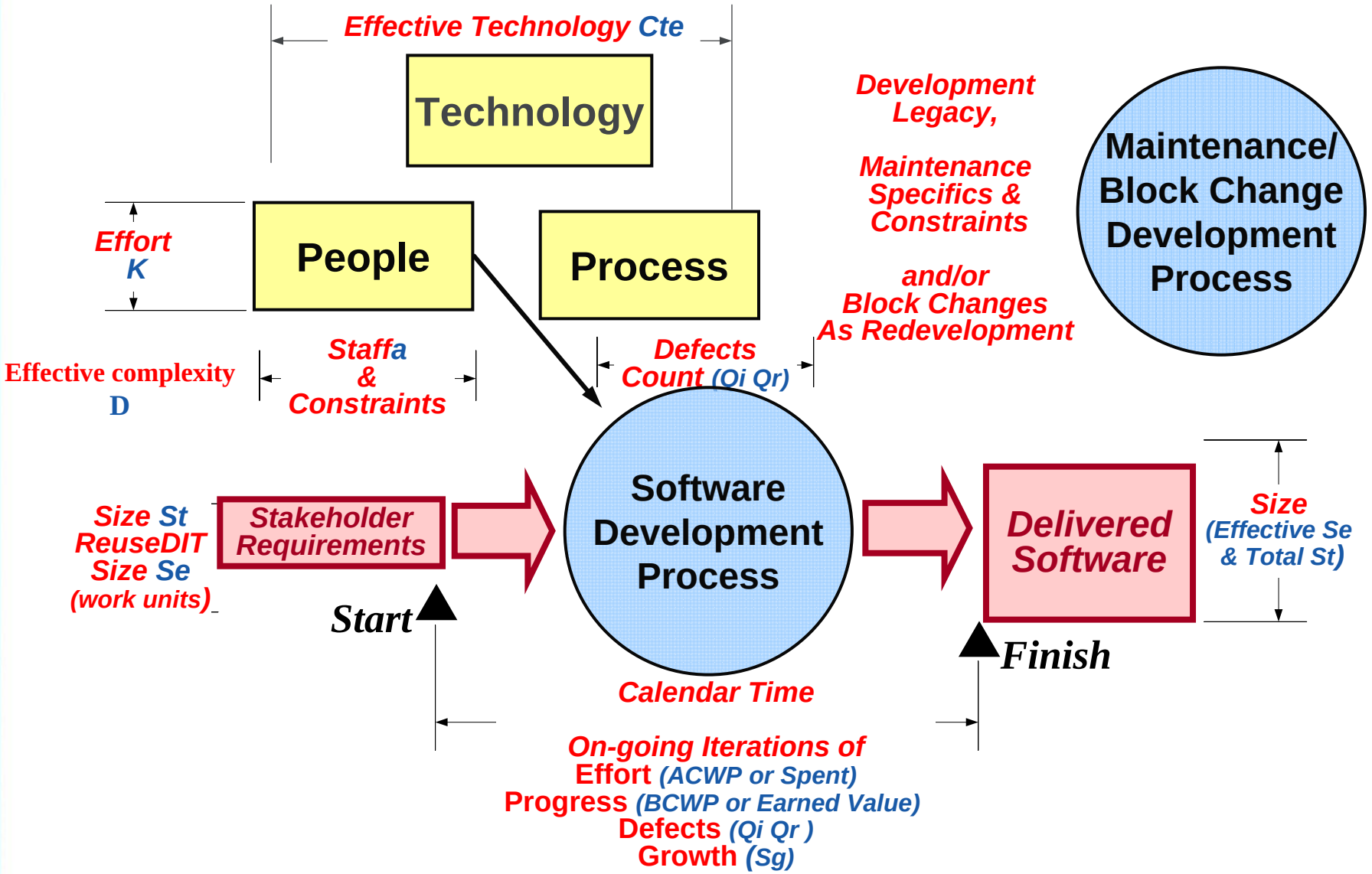
Counts for Pre-existing

Sizing Databases

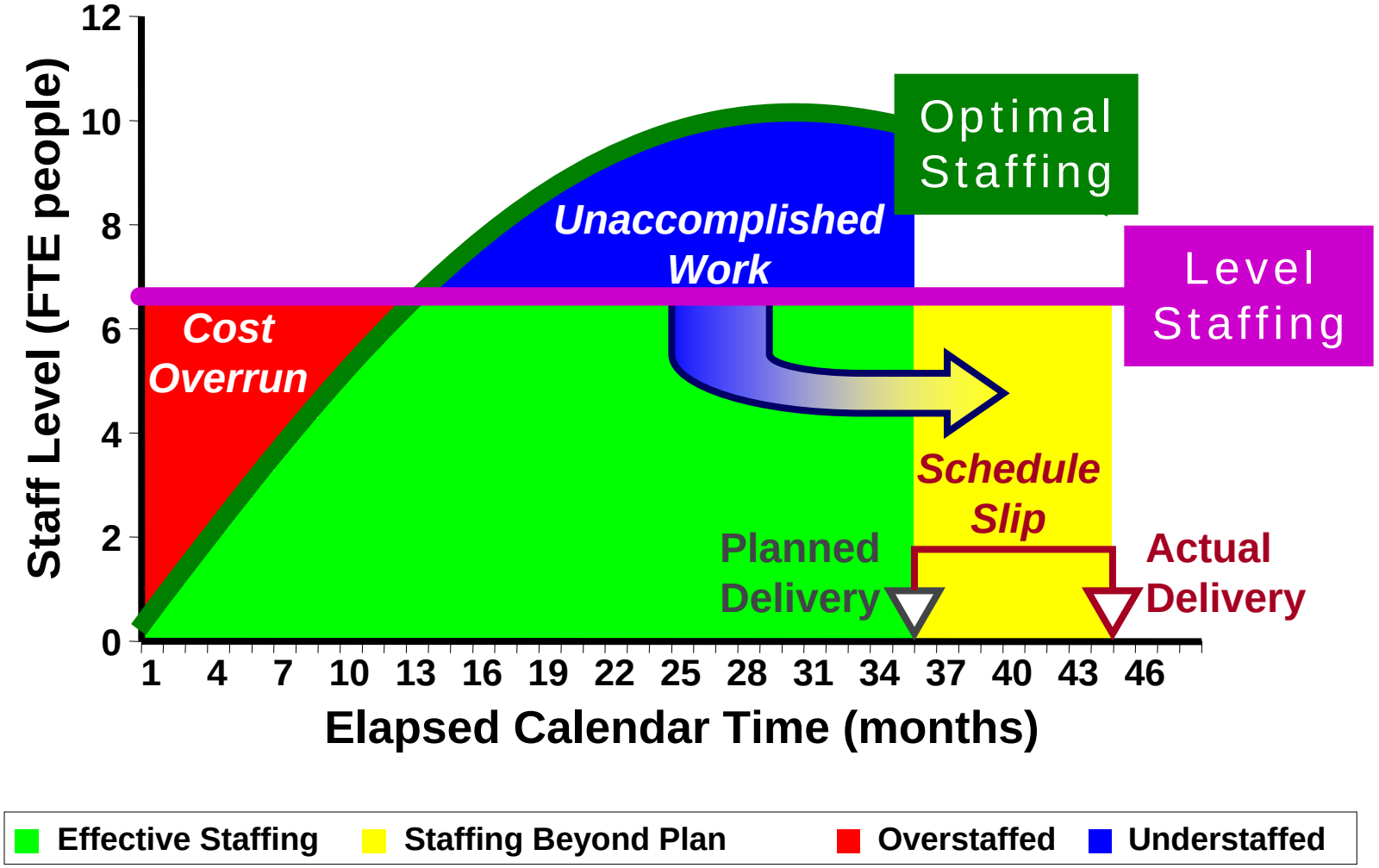
Analogies

…Using Multiple Methods

Software Estimation Basic Model & Associated Metrics



Avoid "Death Marches" and Failed Projects By Applying "Brooks Law"



Step Six: Quantify Risks and Risk Analysis

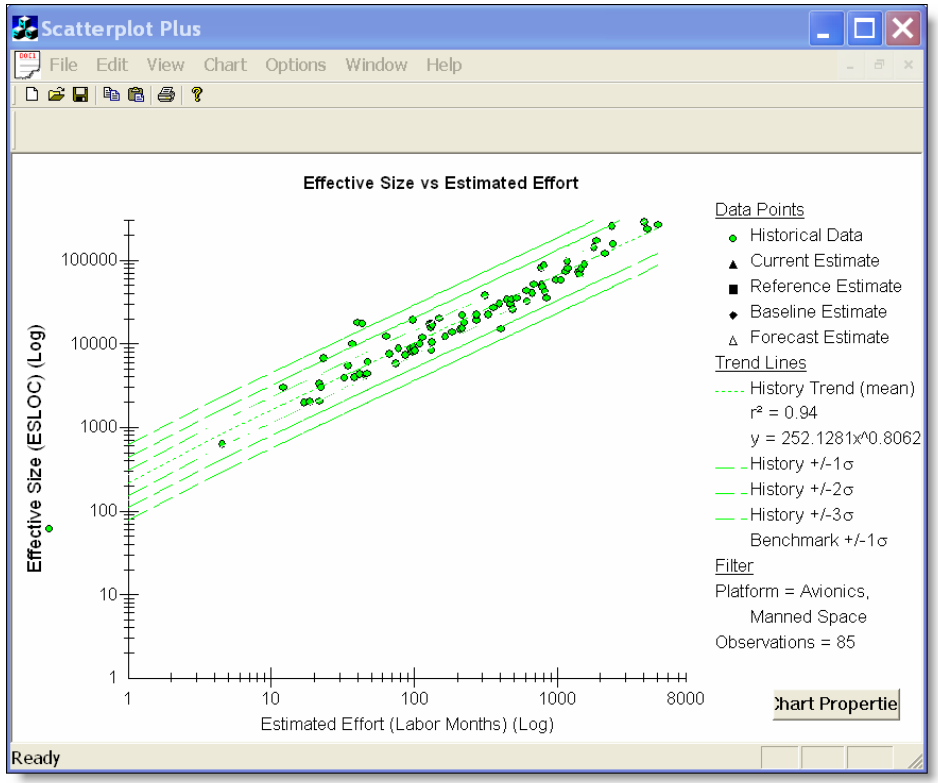
- Risk can produce loss of time, or quality, money, control, understanding…
  - Loss associated with a risk is called the risk impact
- Approximate the probability that the event will occur
  - Risk probability: Likelihood the risk, measured from 0 (impossible) to 1 (certainty) When the risk probability is 1, the risk is a problem since it is certain to happen.
  - Determine how risk can be mitigated
  - Risk control involves a set of actions taken to reduce or eliminate a risk.
- Risk management identifies & addresses internal & external potential threats
  - Problems with sizing and estimating software potentially can have dramatic negative effects
  - If problems can be foreseen & causes acted upon in time, effects can be mitigated
- Although cost, schedule, and product performance risks are interrelated, they can also be analyzed independently
  - Risks must be identified as specific instances in order to be manageable
  - Statistical risk/uncertainty analysis should be a part of schedule & effort estimation process

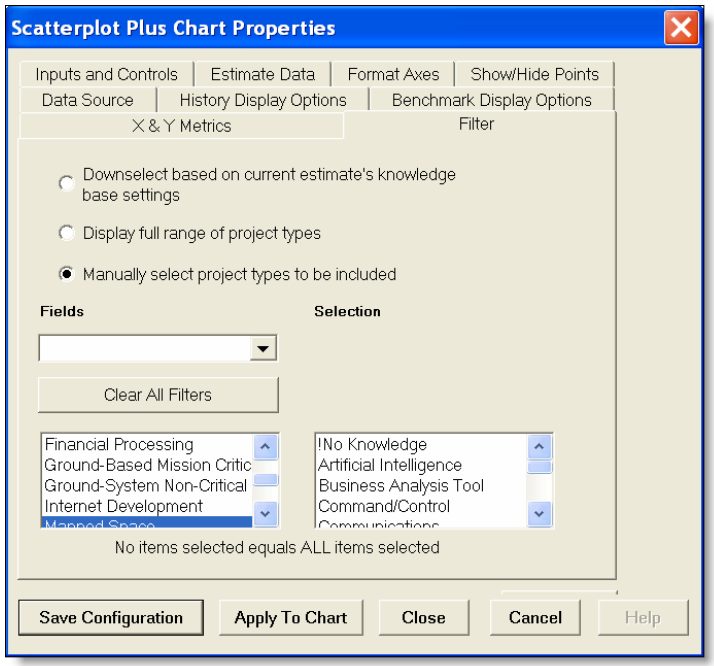
Step Seven: Estimate Validation and Review

- Ideally, validation performed by one who was not involved in generating the estimate
- Assess estimate assumptions
- Ensure groundrules are consistent applied
- Rigorous validation process exposes faulty assumptions, unreliable data and estimator bias
  - Provides clearer understanding inherent risks
  - Isolating problems at source, allows steps to contain associated risks
  - Provides realistic picture of what project will actually require to succeed
- Failing to validate the estimate may result in much greater downstream costs, or even a failed project

Compare Metrics and Sanity Checks

- Shows actual data, ranges, and correlations
- Plots estimates and contrasts with data points
- Plots actual data and / or trends





"In God we trust,
all others bring data."
- W. Edwards Deming

Step Nine: Document Estimate and Lessons Learned

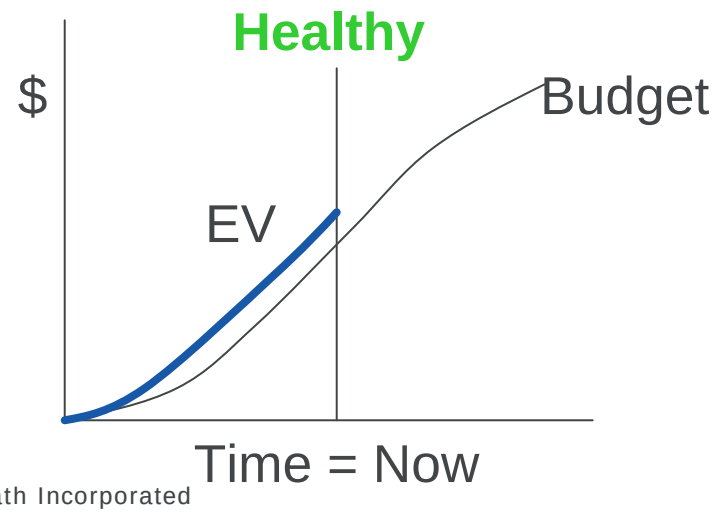
- Document upon estimate complete AND project complete
  - document the pertinent information
  - record the lessons you learned. By doing so, you will have evidence that your process was valid and that you generated the estimate in good faith, and you will have actual results with which to substantiate or calibrate your estimation models.
- Document missing or incomplete information and the risks, issues, and problems that the process addressed and any complications that arose
- Document key decisions made during the estimate & results
- Document dynamics that occurred during the process e.g.
  - interactions of your estimation team
  - interfaces with stakeholders
  - trade-offs made to address issues identified during the process
- Conduct a lessons-learned session
  - As soon as possible after the completion of a project while the participants' memories are still fresh
- Every software project should be used as an opportunity to improve the estimating process

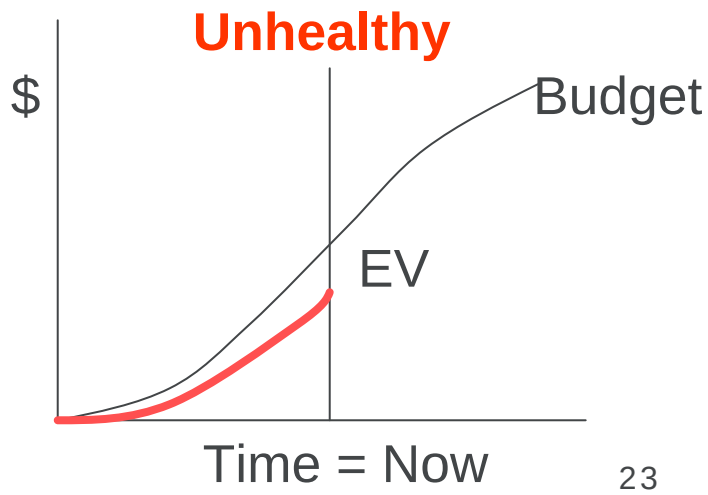
Step Ten: Track Project Throughout Development

- *Refining Estimates throughout Project*
- Once a project has started, use estimates as a basis for performance measurement & project control
- Monitor actual effort & duration of tasks and/or phases
- Evaluate defects & growth in addition to earned value

Use Earned Value TO Quantify Progress Versus Effort

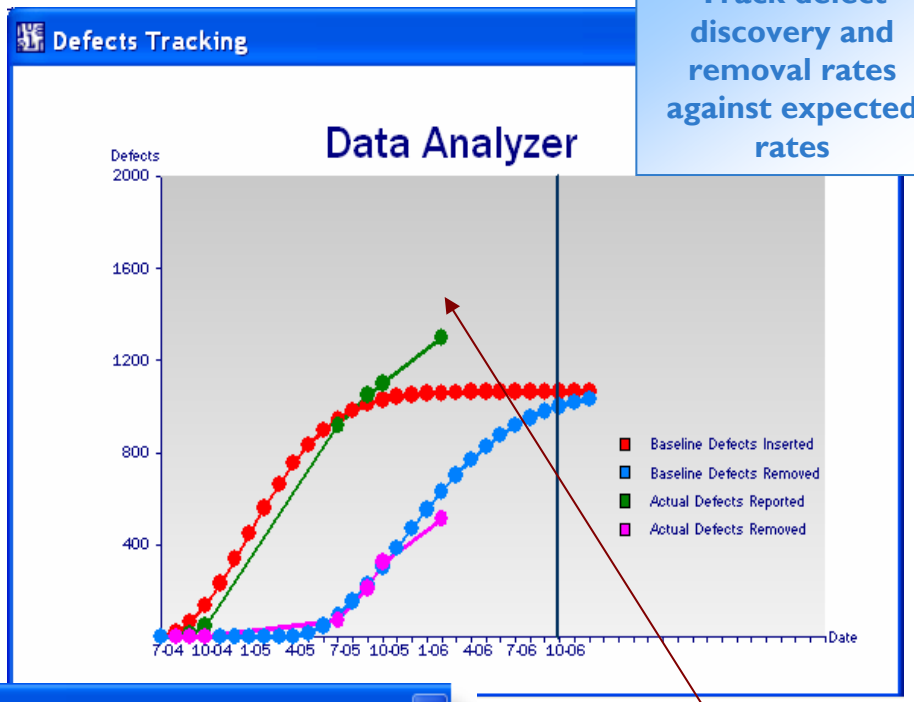
- The main concern of EVM is what has been accomplished in a given time and budget, versus what was planned for the same time and budget
  - A project is generally deemed healthy if what has been accomplished is what was planned, or more
  - A project is deemed unhealthy if accomplishment lags expectations
- Definition: Earned value = budgeted value for the work accomplished (what you got for what it cost you)





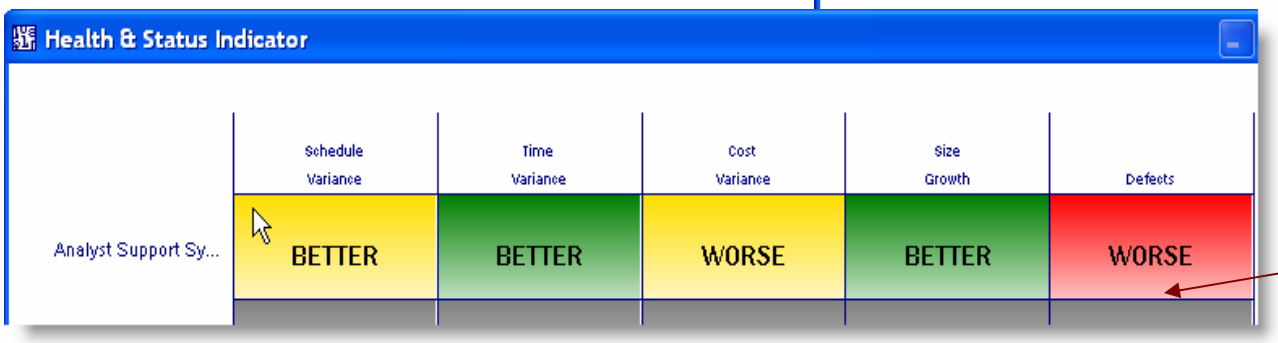
Defects and Growth Impact Software Process

Track defect discovery and removal rates against expected rates



Heath and Status Indicator shows status and trends from the previous snapshot

Thresholds are user definable



Increased defect reporting rate shows a worsening trend

Getting Chaos Projects Under Control

- Identify the “meatballs in the spaghetti”
- Capture and documentation of these units of software
- Clarify subdivisions of work & definitions of “complete”
  - Include reviews as part of complete
- Quickly train team on these processes
- Invoke peer reviews to reduce errors and increase reliability
- Measure and track progress vs effort
- Spend management time where measurement shows issues

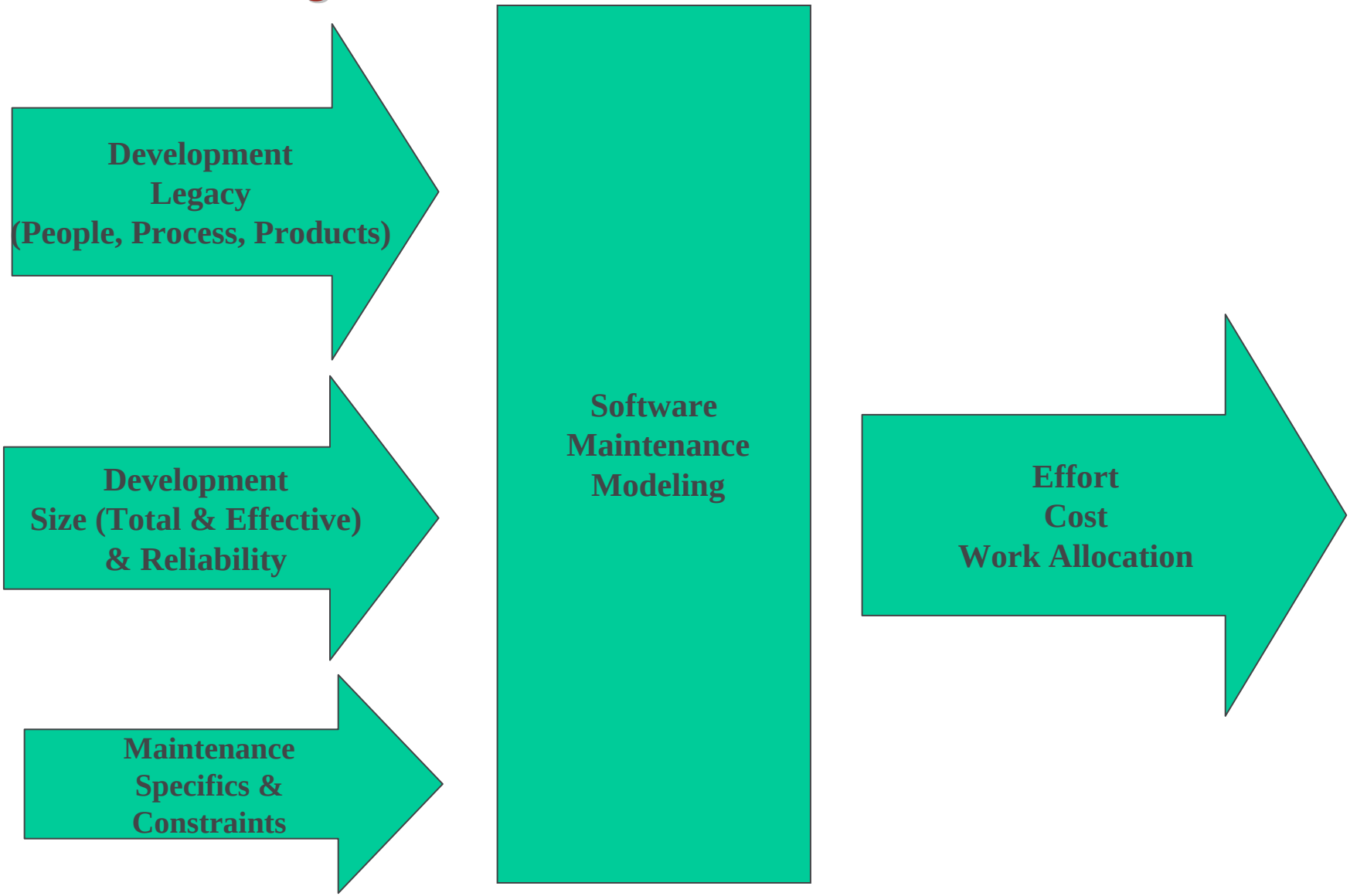
SEER by GALORATH

Maintenance Details

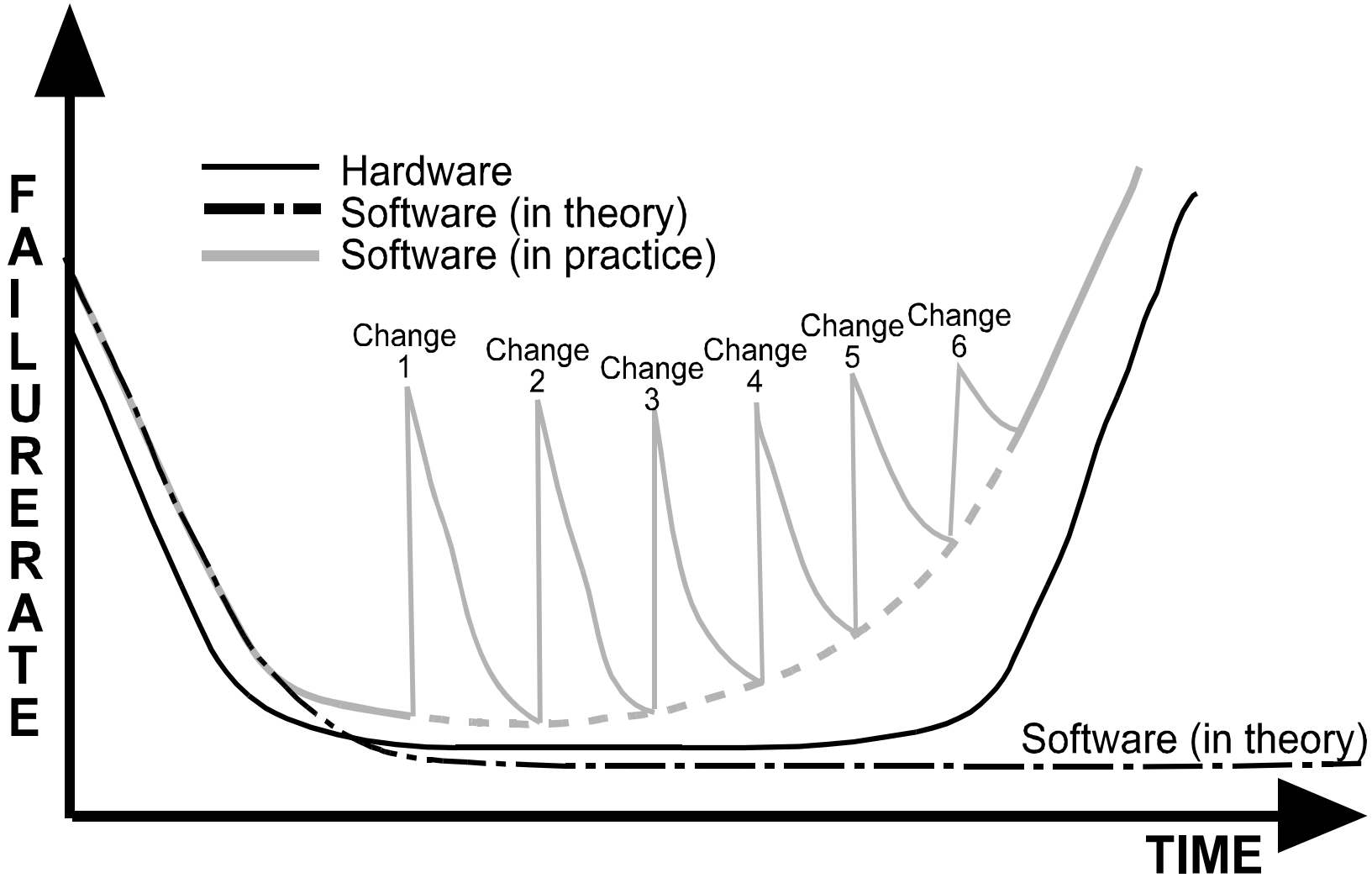
Maintenance Dissected

- Maintenance typically 50% + of the total software workload:
  - Highly dependent on maintenance rigor & operational "life expectancy"
  - Reducing maintenance costs can reduce life cycle costs significantly
- Generally includes sustaining engineering & new function development:
  - Corrective changes (fixing bugs)
  - Adapting to new requirements (OS upgrade, new processor)
  - Perfecting or improving existing functions (improve speed, performance)
  - Enhancing application with (minor) new functions (new feature)
- For every new software product we develop, we get one more to maintain -- for ?? years

Simplified Maintenance Block Diagram



Software Maintenance Is Often A Series of Block Changes



Software Maintenance Goals, Questions, Metrics Adapted from Mitre 1997

Goal	Question	Metric(s)
Maximize Customer Satisfaction	How many problems affect the customer?	1. Current Change Backlog 2. Software Reliability
Minimize cost	How much does a software maintenance delivery cost?	
	How are costs allocated	Cost per activity
	What kinds of changes are being made?	Number of changes by type
	How much effort is expended per change	Staff hours expended by change /type
Minimize Schedule	How difficult is the delivery?	Complexity Assessment Software Maintainability Computer resource Utilization
	Are we meeting delivery schedules?	Percentage of On-Time Deliveries

Development Quality Impacts Maintenance

http://www.bcs.org/server.php?show=ConWebDoc.3063

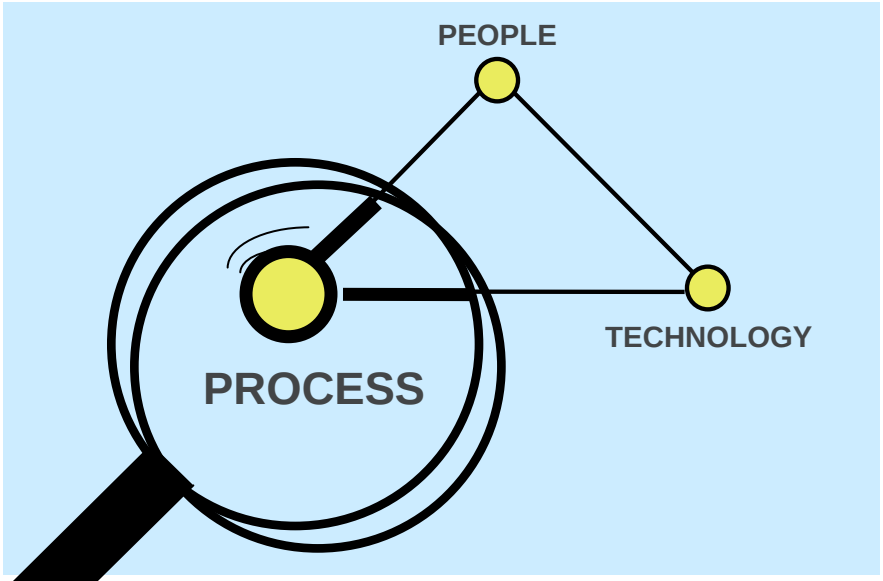
- IEEE Std 1919-1993: Software maintenance defines maintenance as:

  Modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment

  - States that maintenance starts after delivery
- **Largest costs of software production occur after the 'development phase' is complete**
  - Maintenance up to 75 per cent of the total ownership cost.
- **Maintenance costs generally not result of poor requirements or design**
- **Even if "right the first time" change is inevitable:**
  - Political decisions (e.g. introduction of a new tax).
  - Hardware related changes.
  - Operating system upgrades over time.
  - Competition - new features to be added.
  - System almost instantly complying to outdated requirements
- **Construction may not affect function, but greatly affects future maintainability**
- **Maintainability goals during development can significantly reduce total ownership costs**

Major Process Improvement Goal Lowering Defects. Source CMMI Tutorial

- Everyone realizes the importance of having a motivated, quality work force but...
- ...even our finest people can't perform at their best when the process is not understood or operating "at its best."



Major determinants of product cost, schedule, and quality

Software Maintenance Critical Success Factors

- Functionality: Preserve or enhance functionality
- Quality: Preserve or increase quality of system
- Complexity: Should not increase product complexity relative to the size
- Volatility: should not lead to increase in product volatility
- Costs: Relative costs per maintenance task should not increase for similarly scoped tasks
- Deadlines: Agreed upon release deadlines should be kept and delays should not increase
- User Satisfaction: Increase or at least not decrease
- Profitability: Be profitable or at least cover its costs

Why Maintenance Is Hard

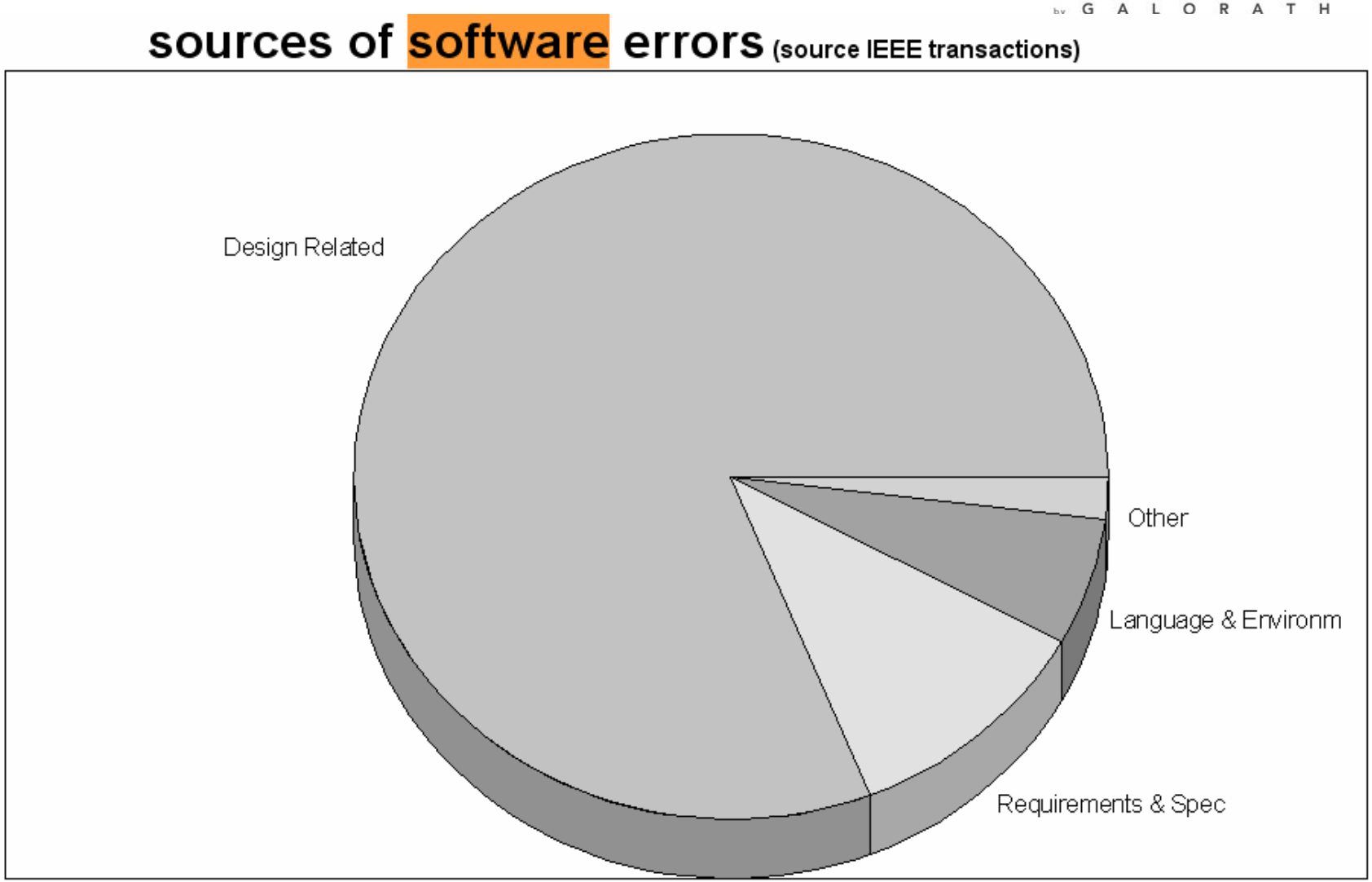
- May not have had maintenance as a goal
- System may not have been fully tested
- Documentation may be inadequate
- Maintenance staff may be inexperienced
- The tendency to produce quick & dirty fixes
- Process or language experience may have left a mess
- The "but I only changed 1 line syndrome"

Why Software Maintenance Costing Is Harder

- Software Maintenance treated as A Level Of Effort Activity
- This Means You Can Maintain Software With A Larger Or Smaller Staff Depending On Your Desires / Budget

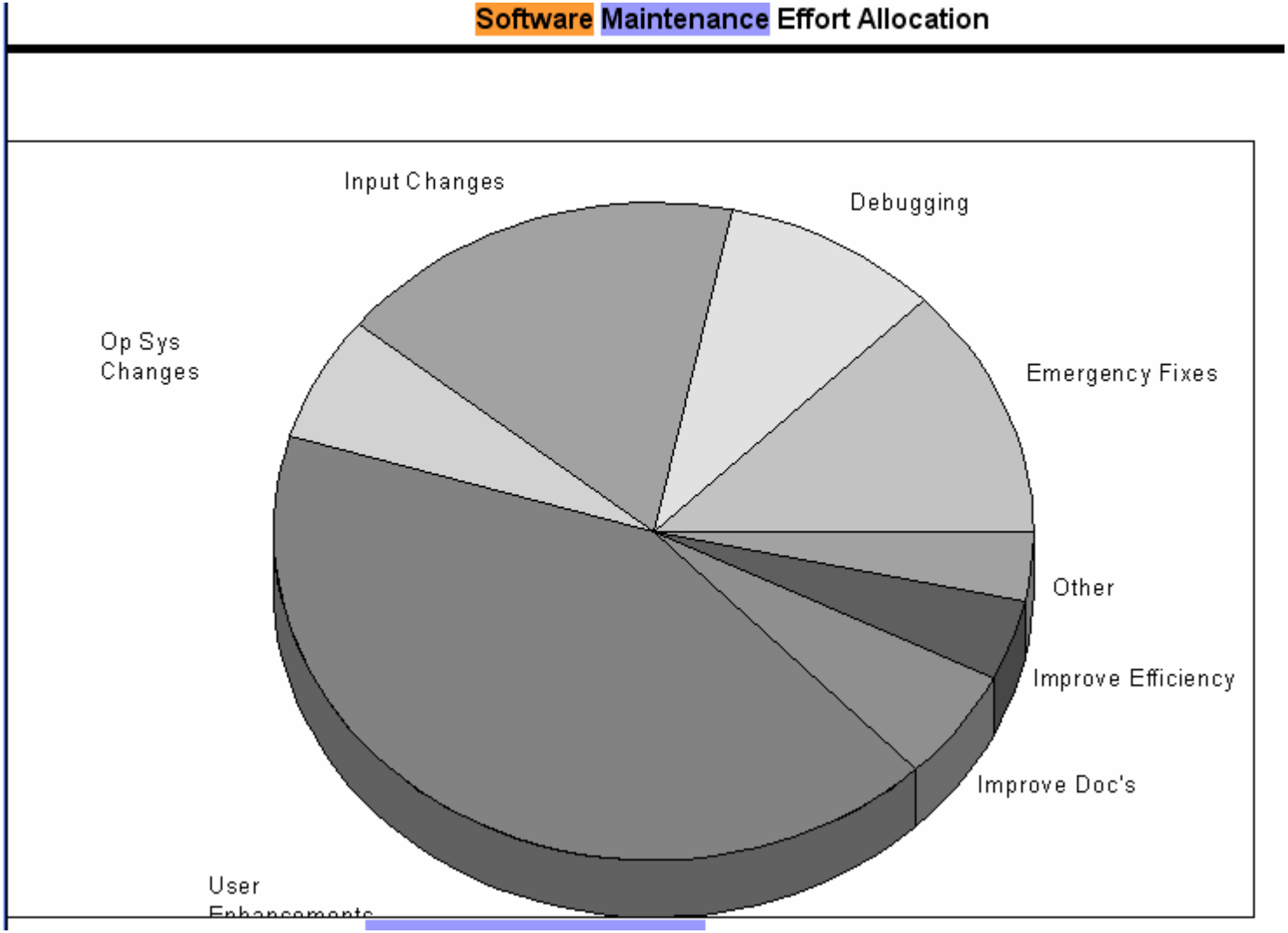
Maintaining A Car	Maintaining Software
High Maintenance: **Go By The Book (Regular Oil Changes, Etc.)**	• **Fix emergencies** • **Provide new functionality as needed** • **Adapt as necessary** • **Software may not degenerate over time**
Nominal Maintenance: **Go Partially By The Book (Less Frequent Oil Changes, Etc.**	• **Fix emergencies** • **Provide some required new functionality** • **Adapt when there is time**
Low Maintenance: **Go Slightly By The Book (Add Oil When The Low Oil Light Goes On**	• **Fix only emergencies and small adaptations** • **Software will degenerate over time**

Sources of Software Errors



Software Maintenance Effort Allocation

Allocation of Softwaree Effort Source IEEE



Development Defects Analysis Is a Clue to Maintenance Issues

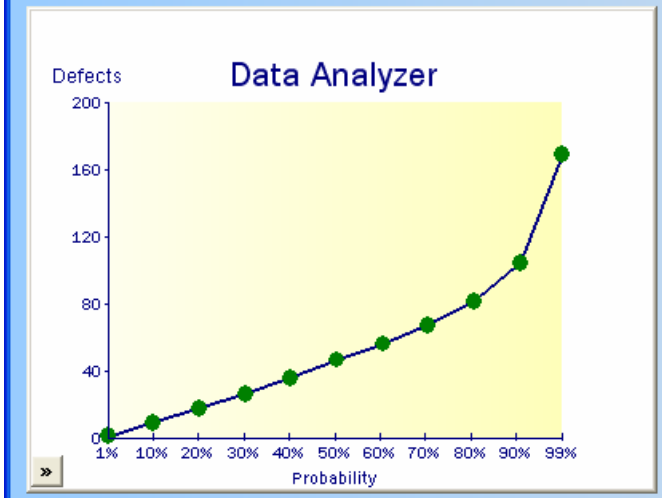
SEER by GALORATH

Defects Analysis - Program: Data Analyzer

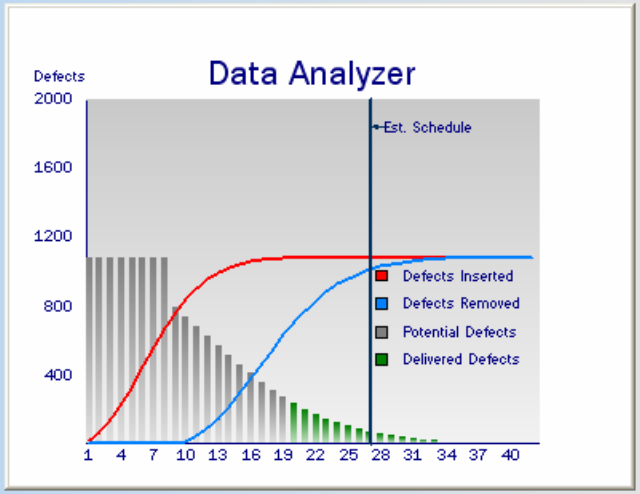
Time Phased Defects

Months From Estimate	Delivery Date	Hours	Est. Cost	Delivered Defects	Defect Density	Cost Difference	Marginal Cost / Defect Removed
-8	6/30/08	28,330	3,187,117	268	7.68	-2,669,728	
-7	7/30/08	31,121	3,501,165	230	6.61	-2,355,680	8,418
-6	8/30/08	33,996	3,824,578	197	5.65	-2,032,267	9,620
-5	9/30/08	36,938	4,155,528	167	4.79	-1,701,316	11,033
-4	10/30/08	39,930	4,492,138	140	4.03	-1,364,707	12,701
-3	11/30/08	42,956	4,832,523	117	3.36	-1,024,322	14,678
-2	12/30/08	45,998	5,174,829	97	2.78	-682,015	17,029
-1	1/30/09	49,042	5,517,264	80	2.29	-339,581	19,838
Estimate	3/02/09	52,061	5,856,845	65	1.87	0	23,120
1	3/30/09	55,073	6,195,760	53	1.51	338,916	27,366
2	4/30/09	58,033	6,528,697	42	1.21	671,853	32,171
3	5/30/09	60,938	6,855,538	34	0.97	998,694	38,131
4	6/30/09	63,778	7,175,022	27	0.76	1,318,177	45,400
5	7/30/09	66,542	7,486,020	21	0.60	1,629,175	54,304
6	8/30/09	69,223	7,787,538	16	0.47	1,930,694	65,255

Defects Risk



Defect Profile



Maintenance Drivers: Scope

- Years of Maintenance
  - Number of years for which software maintenance costs will be estimated
    - Maintenance typically begins when operational test & evaluation is completed
- Separate Sites
  - Number of separate operational sites where the software will be installed and users will have an input into system enhancements
    - Count only sites that have some *formal* input
    - Do not necessarily count all user sites
  - Alters both amount and allocation of maintenance effort
    - More sites = more enhancing, corrective, and perfective effort

Maintenance Growth Over Life

- Anticipated size growth from the point immediately after the software is turned over to maintenance to the end of the maintenance cycle
- May include additions of new functionality

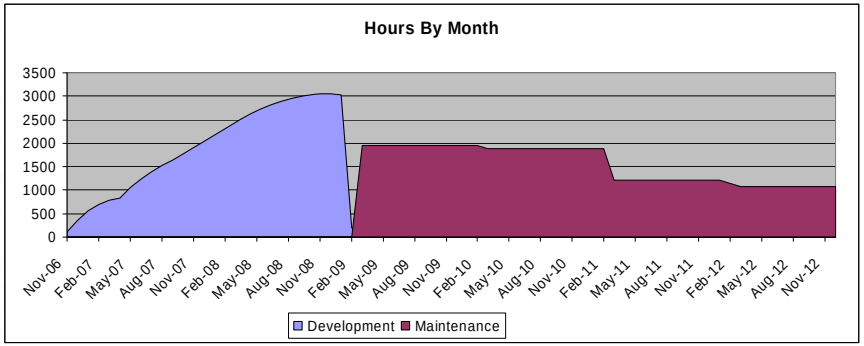
Rating	Description
100%	Major updates adding many new functions
35%	Moderate updates adding some new functions
20%	Minor updates & enhancements to existing functions
5%	No updates expected, some minor enhancements
0%	Sustaining engineering only

0 vs 100% growth over 5 years

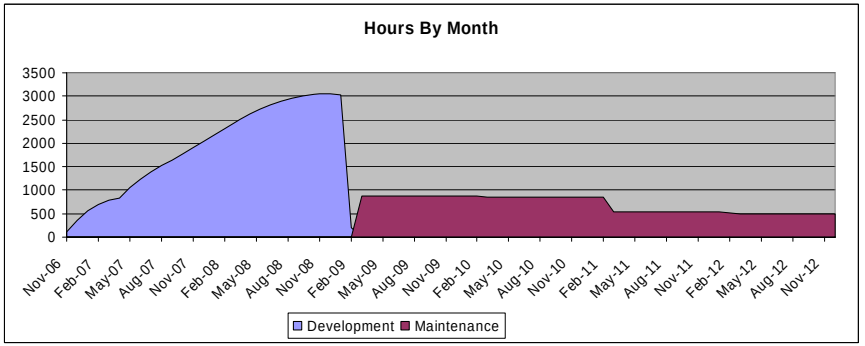
Quick Estimate

	Program: Data Analyzer Estimate	Program: Data Analyzer Reference	Diff.
Development Schedule Months	27.07	27.07	0%
Development Effort Months	342.51	342.51	0%
Development Effort Hours	52,061	52,061	0%
Development Base Year Cost	5,856,845	5,856,845	0%
Maintenance Effort Months	584.23	260.59	124%
Defect Prediction	65	65	0%
Constraints	MIN TIME	MIN TIME	

100% growth over 5 years
Initial 27 mo development



0% growth over 5 years
Initial 27 mo development



Technology & Environment Differences

- **Personnel Differences**
  - Rates maintenance personnel's capabilities and experience in comparison to development personnel
- **Development Environment Differences**
  - Rates the quality of the maintenance environment in comparison to the tools and practices used in the development environment

Rating	Description
Very High	Significantly better than development
High	Slightly better than development
Nominal	Same as development
Low	Somewhat less than development
Very Low	Significantly lower than development

Annual Change Rate

- Average percent of the software impacted by software maintenance and sustaining engineering per year
- May include changes, revalidation, reverse engineering, redocumentation, minor changes for new hardware, or recertification

Rating	Description
35%	Very High
15%	High
11%	Nominal
5%	Low
0%	Very Low

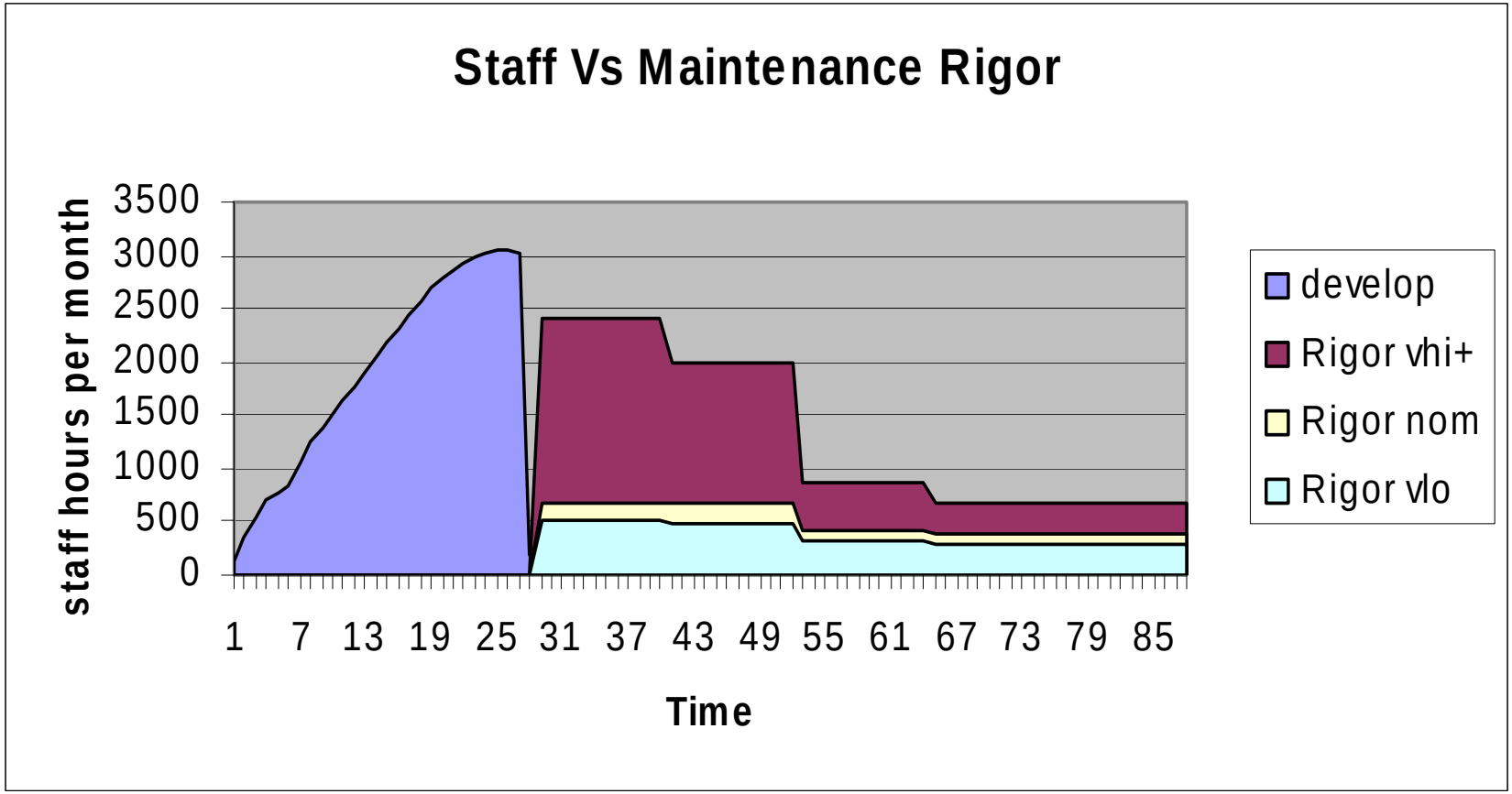
50% vs 0 annual change over 5 years

Quick Estimate

	Program: Data Analyzer Estimate	Program: Data Analyzer Reference	Diff.
Development Schedule Months	27.07	27.07	0%
Development Effort Months	342.51	342.51	0%
Development Effort Hours	52,061	52,061	0%
Development Base Year Cost	5,856,845	5,856,845	0%
Maintenance Effort Months	392.21	282.65	39%
Defect Prediction	65	65	0%
Constraints	MIN TIME	MIN TIME	

Key Driver: Maintenance Level (Rigor)

Most Projects Spend Low During Maintenance



Percent to be Maintained

- Enter the percent of the total code that will be maintained
- If maintenance will be shared with another organization, enter only the portion to be included in this estimate
- If software cannot be changed, do not include it in the percent to be maintained (e.g. non updateable embedded processors)

Rating	Description
100%	Maintenance for entire WBS element will be included in the estimate
15%	Maintenance effort is outside the estimate, but some maintenance integration effort is required
0%	No maintenance effort is included in the estimate

Maintain Total System

- Parameter determines whether *total* size or *effective* size should be used to estimate maintenance
  - If the software is entirely new lines of code, this parameter has no effect
- Default setting is YES so that maintenance is estimated based on the entire completed Program, not just the changes
  - Set to **NO** if preexisting code maintenance is someone else's responsibility
- For COTS, this should be set to NO since you generally don't maintain the COTS package (the vendor does this)

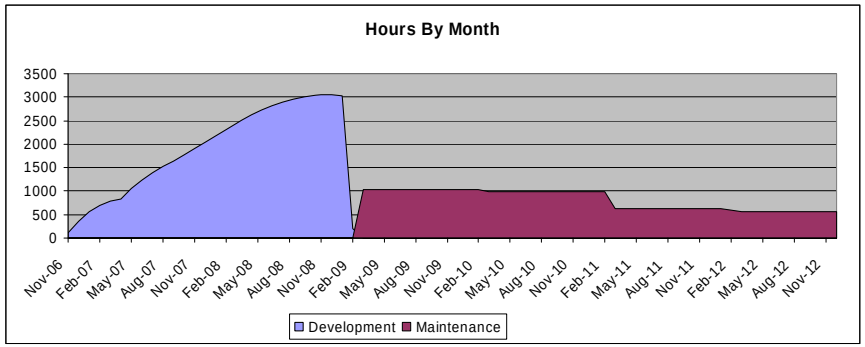
Rating	Description
YES	Normal: Estimate maintenance of the total WBS element, including preexisting code
NO	Special: Estimate maintenance of the effective size (current changes) only. Maintenance of the entire preexisting code is not included in the estimate

Steady State Maintenance Only

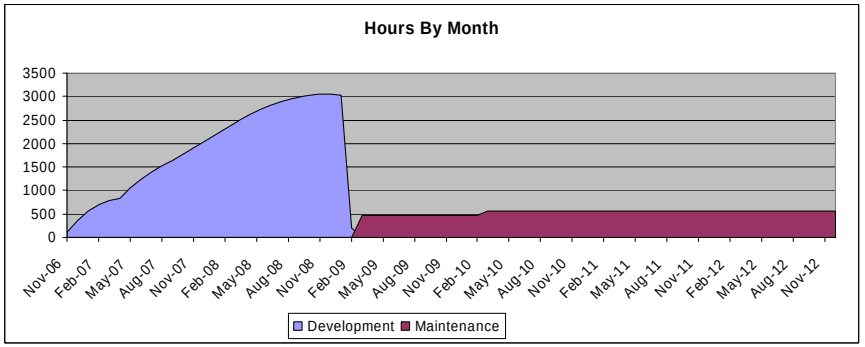
- Indicates whether maintenance profile should be effort-based, or fixed staff.

Rating	Description
YES	Estimate maintenance with a fixed annual staff level. (For Contracts where level of effort will not allow rampdown or planned initial block change will be added to effort)
NO	Estimate maintenance with additional effort in the first years.

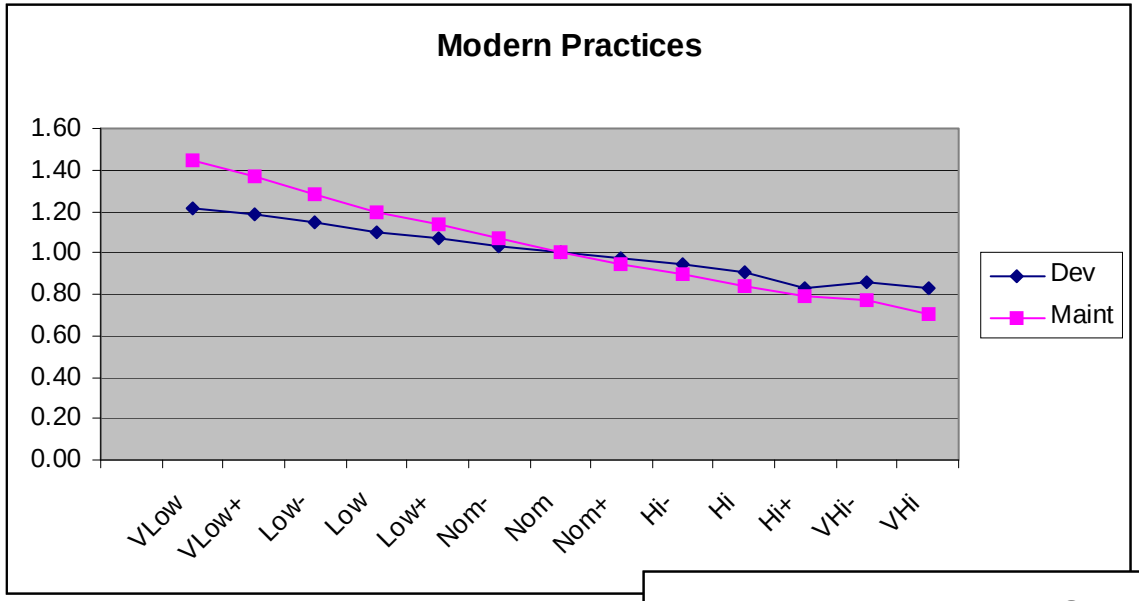
no

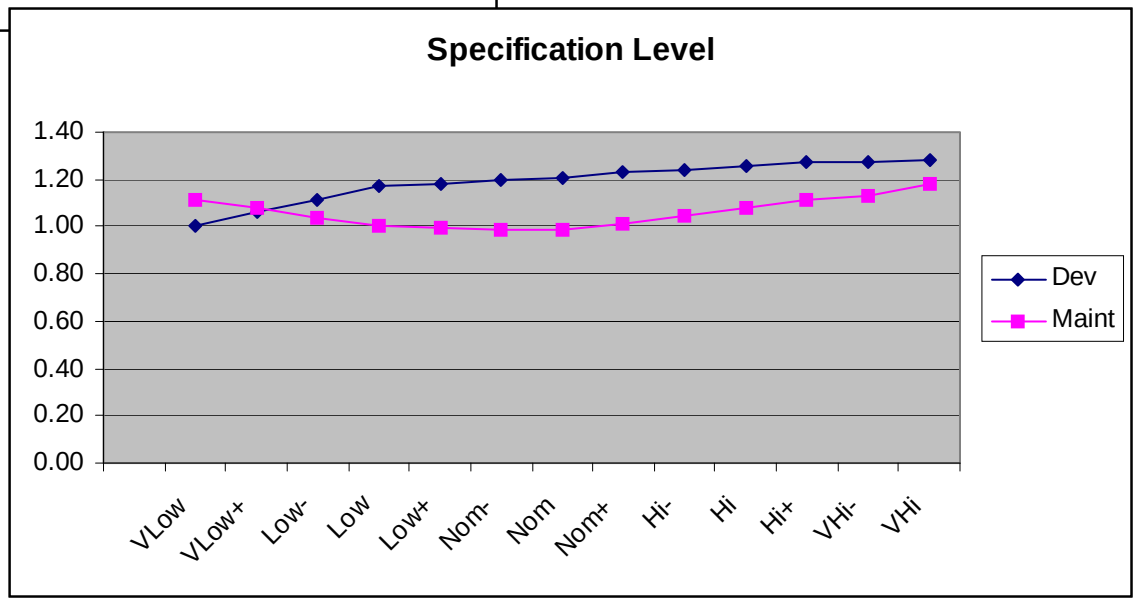


yes

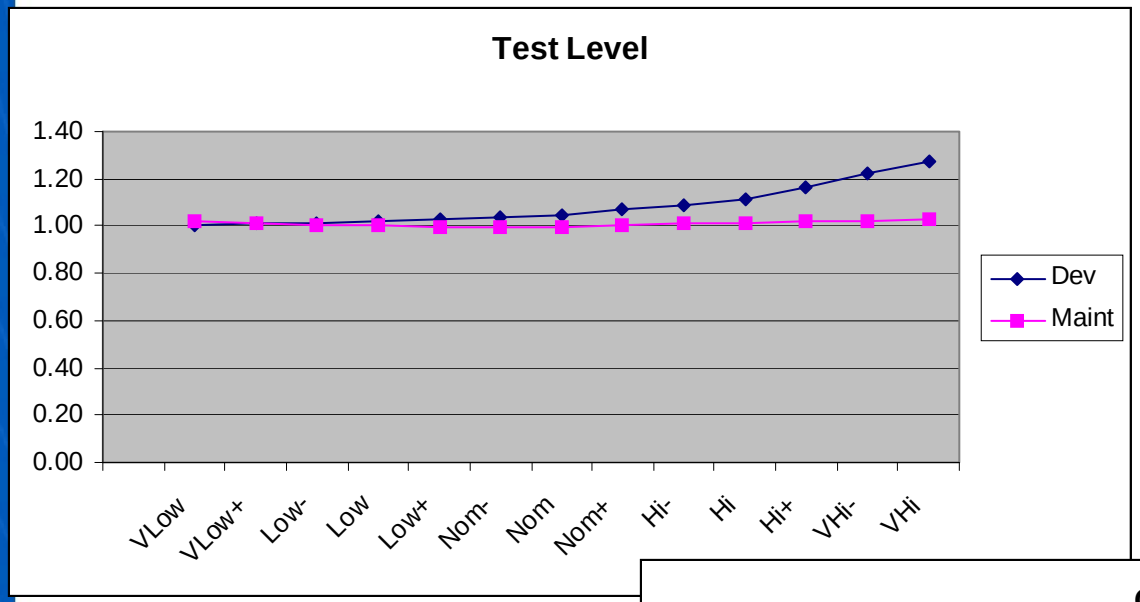


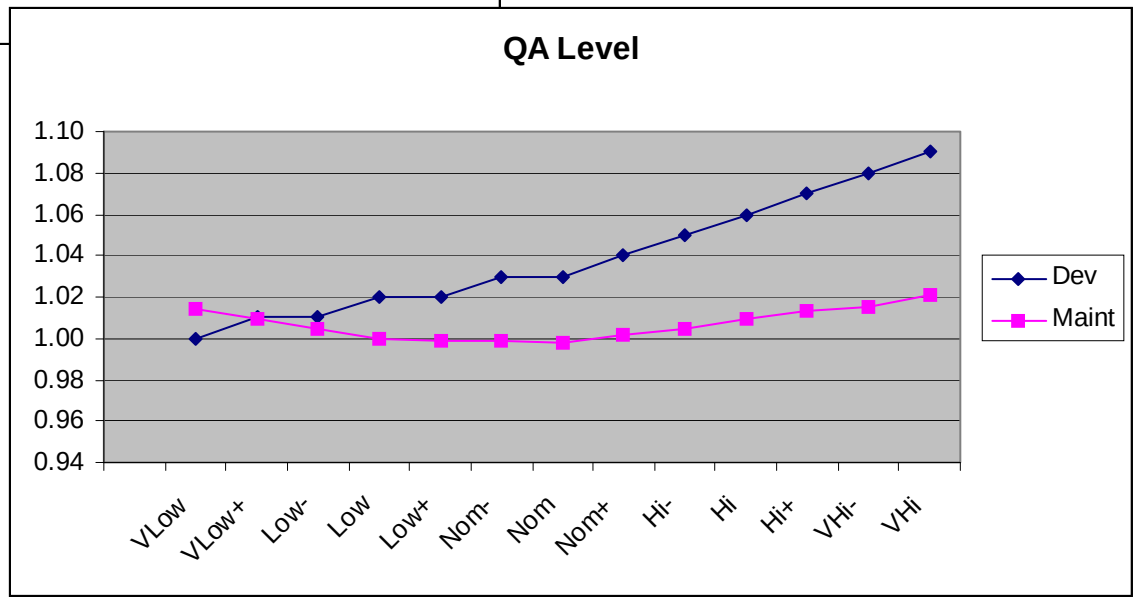
People, Process, Technology Sensitivity Development Vs Maintenance – 1



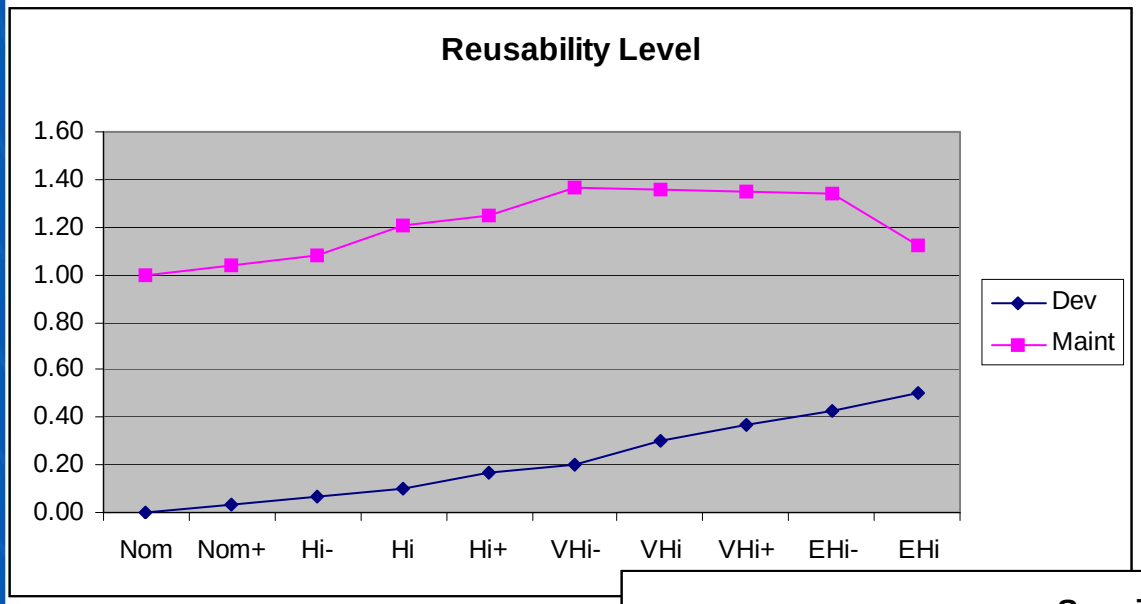


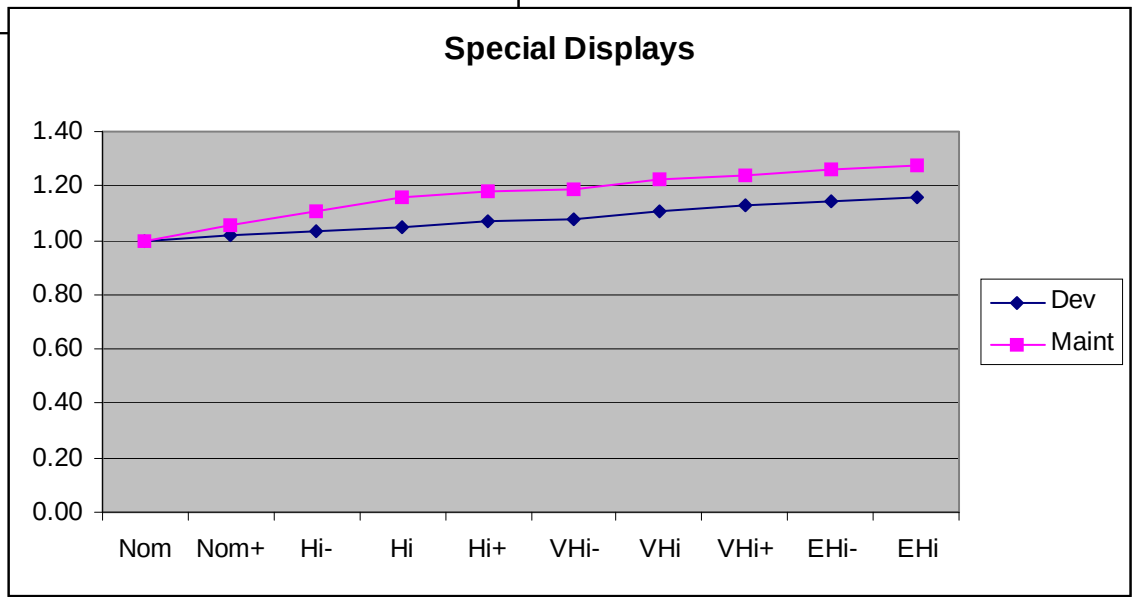
Development Vs Maintenance - 2





Development Vs Maintenance - 3





Defects Can Be Reduced By Further Development Testing but Not Eliminated

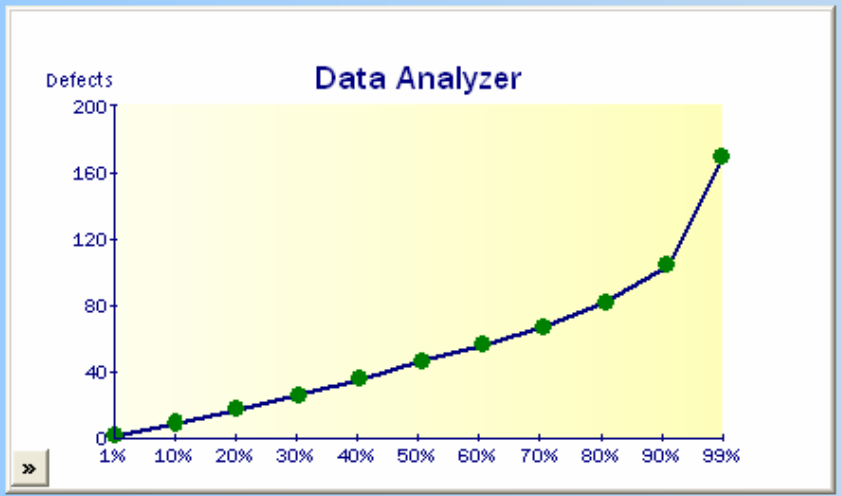
SEER by GALORATH

Defects Analysis - Program: Data Analyzer

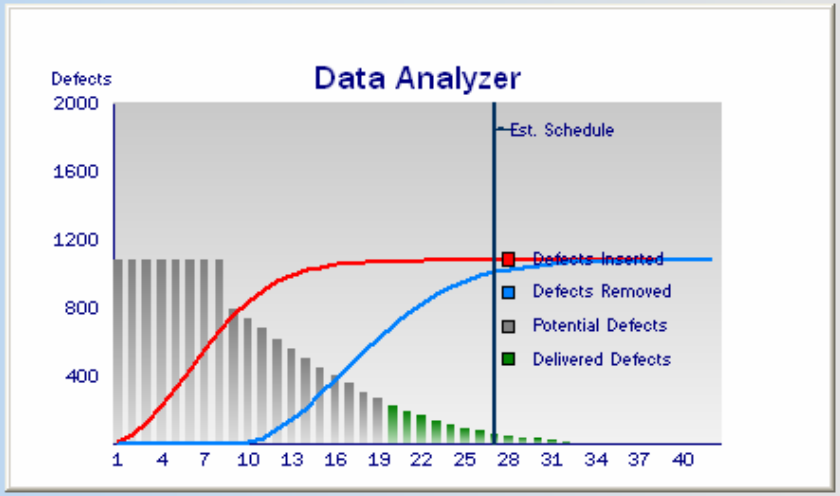
Time Phased Defects

Months From Estimate	Delivery Date	Hours	Est. Cost	Delivered Defects	Defect Density	Cost Difference	Marginal Cost / Defect Removed
-8	7/01/08	28,330	3,187,117	268	7.68	-2,669,728	
-7	7/31/08	31,121	3,501,165	230	6.61	-2,355,680	8,418
-6	8/31/08	33,996	3,824,578	197	5.65	-2,032,267	9,620
-5	10/01/08	36,938	4,155,528	167	4.79	-1,701,316	11,033
-4	10/31/08	39,930	4,492,138	140	4.03	-1,364,707	12,701
-3	12/01/08	42,956	4,832,523	117	3.36	-1,024,322	14,678
-2	12/31/08	45,998	5,174,829	97	2.78	-682,015	17,029
-1	1/31/09	49,042	5,517,264	80	2.29	-339,581	19,838
Estimate	3/03/09	52,061	5,856,845	65	1.87	0	23,120
1	3/31/09	55,073	6,195,760	53	1.51	338,916	27,366
2	5/01/09	58,033	6,528,697	42	1.21	671,853	32,171
3	5/31/09	60,938	6,855,538	34	0.97	998,694	38,131
4	7/01/09	63,778	7,175,022	27	0.76	1,318,177	45,400

Defects Risk



Defect Profile



7 Characteristics of a Dysfunctional Software Projects

(Source: Mike Evans, et al.)

- Failure to Apply Essential Project Management Practices
- Unwarranted Optimism and Unrealistic Management Expectations
- Failure to Implement Effective Software Processes
- Premature Victory Declarations
- Lack of Program Management Leadership
- Untimely Decision-Making
- Lack of Proactive Risk Management

Conclusions

- Software Maintenance can be 75% of total ownership costs
- Development decisions, processes and tools can impact maintenance costs
- Generally even a perfect delivered system quickly needs upgrade
- While software maintenance is often treated as a level of effort activity there are consequences:
  - Quality, functionality and reliability
- Software total ownership costs and risks can be estimated using SEER for Software