estimate

estimate • analyze • plan • control

# Total Cost of Ownership: Development is (Only) Job One

Dan Galorath

galorath@galorath.com

310 414-3222 x614

Blog: www.galorath.com/wp

SEER by GALORATH

# Total Cost of Ownership: Development is (Only) Job One


- **Abstract:**

- Planning software development projects is never an easy undertaking. Customer and competitive requirements, time-to-market, architectural and quality considerations, staffing levels and expertise, potential risks, and many other factors must be carefully weighed and considered. What can make software planning even more complicated, however, is that software development costs only comprise a portion – often the smaller portion – of the total cost of software ownership. In fact, the development process, itself, invariably has a significant impact on total cost of ownership as tradeoffs are evaluated and compromises made which impact software sustainability and maintainability of software over time.
- Because software doesn't wear out like car tires do, software planners may underestimate how much a code stream can degrade over time with the accumulation of patches, system and configuration changes, provisioning and re-provisioning, integrations, and ongoing development. Further, the rigorous standards applied during initial software development may end up being compromised as maintenance personnel are diverted to emerging or mission-critical software issues. Over time, accumulation of poorly managed changes almost always generates software instability and a significant increase in the cost of software maintenance – up to four times the cost of initial development, according to some estimates.
- This session will provide a systematic approach to addressing total cost of ownership across the software lifecycle, including design for maintainability, development of measurement criteria, collection of metrics, and industry standards, guidelines, and best practice options. Parametric modeling will be discussed using the SEER platform as a specific example of this approach. Estimating block changes and their potential interdependencies and impacts will also be covered.

estimate

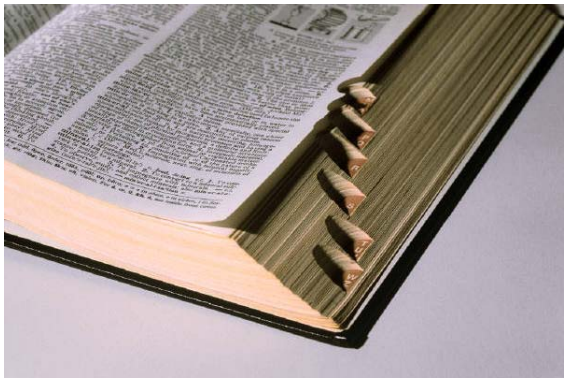
estimate • analyze • plan • control

# Estimation & Estimation Process

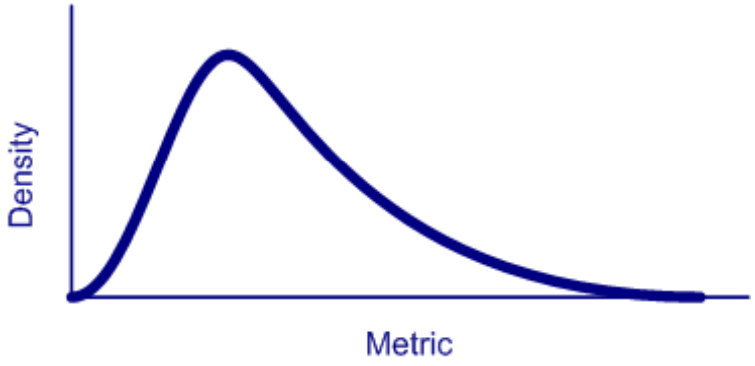
SEER by GALORATH

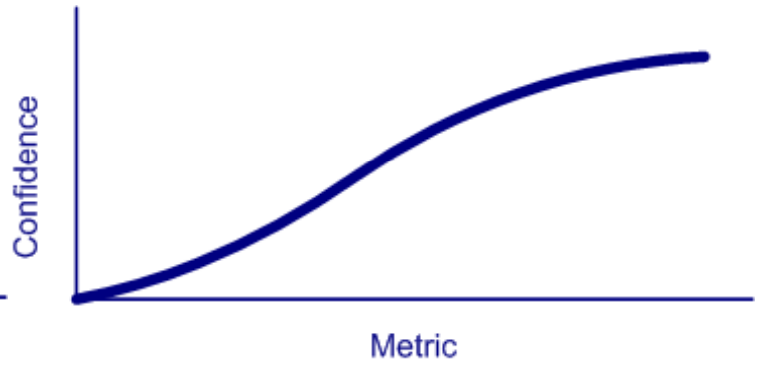
# An Estimate Defined

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



- Estimates more precise with progress
- *A WELL FORMED ESTIMATE IS A DISTRIBUTION*



# SEER Addresses Many Top CIO Business Priorities

## Analysts

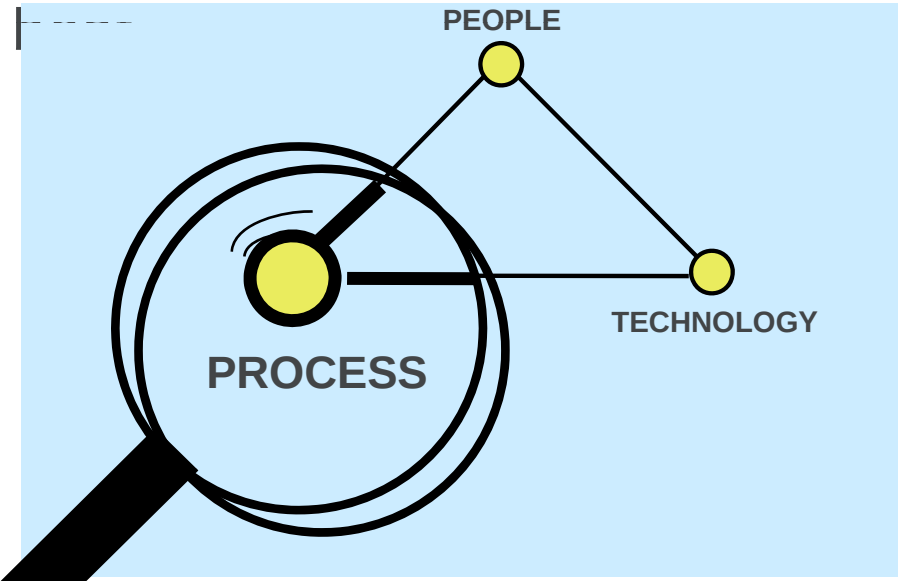
1. Business process improvement
2. Controlling enterprise-wide operating costs
3. Attract, retain and grow customer relationships
4. Improve effectiveness of enterprise work force
5. Revenue growth
6. Improving competitiveness
7. Using intelligence in products and services
8. Deploy new business capabilities to meet strategic goals
9. Enter new markets, new products or new services
10. Faster innovation

## CIO Insight.com

1. Service to customers
2. Improving business processes
3. Contributing to business strategy creation
4. Cutting costs
5. Innovative products & services
6. Creating more business
7. Improving workforce productivity
8. Ensuring business continuity
9. Complying with regulatory requirements
10. Differentiating my company from competitors thru IT

# People, Process, Technology Are Keys Source CMMI Tutorial

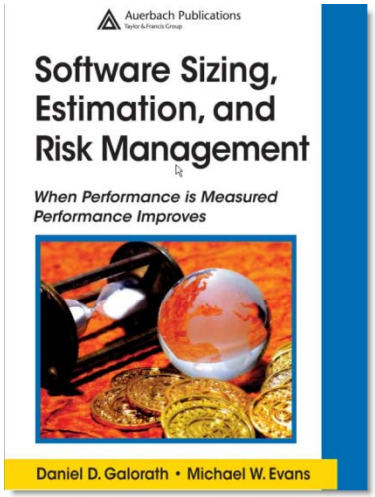
- Everyone realizes the importance of having a motivated, quality work force but…
- …even our finest people can't perform at their best when the process is not understood or operating "at its




**Major determinants of product cost, schedule, and quality**

# 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

# estimate

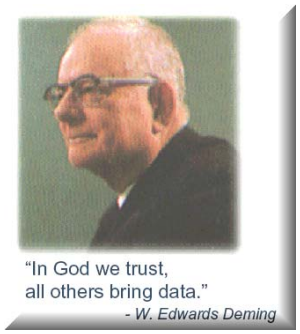
estimate • analyze • plan • control

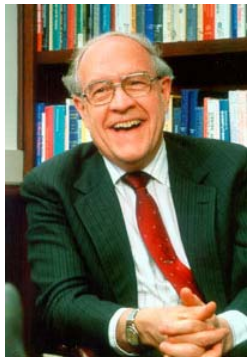
## Software Measurement

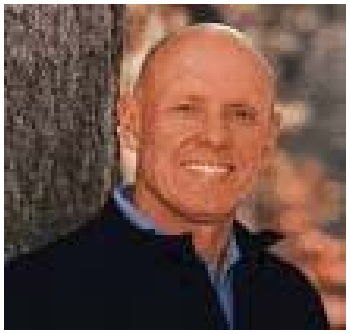
SEER by GALORATH

# Some Measurement Heroes

- 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
- Eli Goldratt: Improvements should increase profit Effectiveness

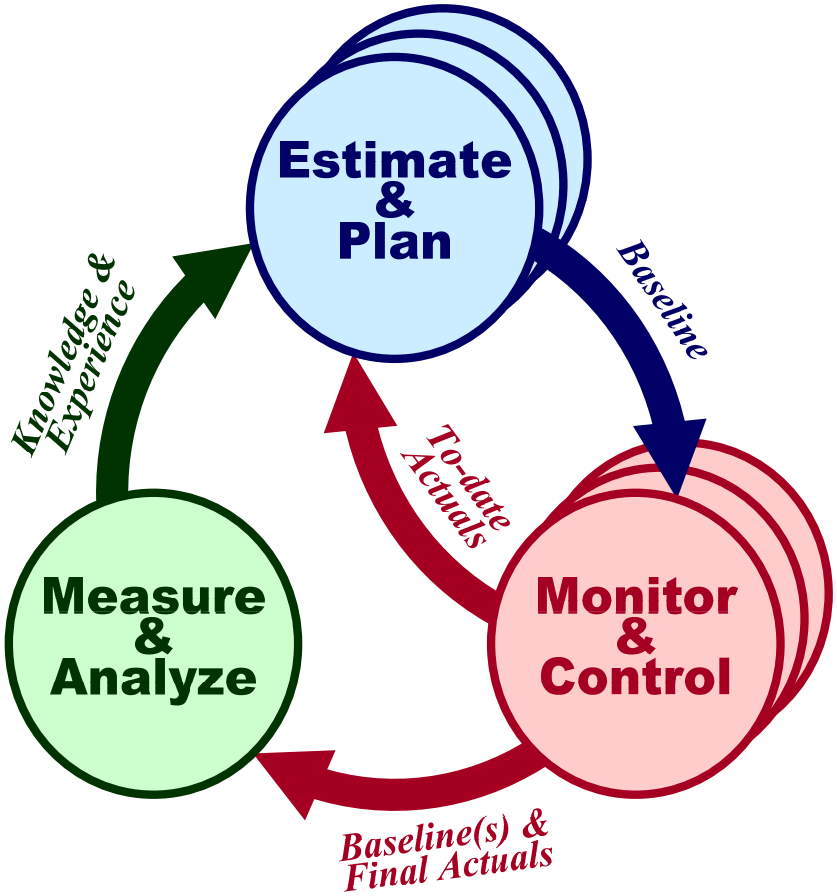


# Reasons For Measurement

- Measure To Improve
- Measure To Estimate
- Measure To Benchmark
- Measure To Assess

# Supporting Successful Projects With Process

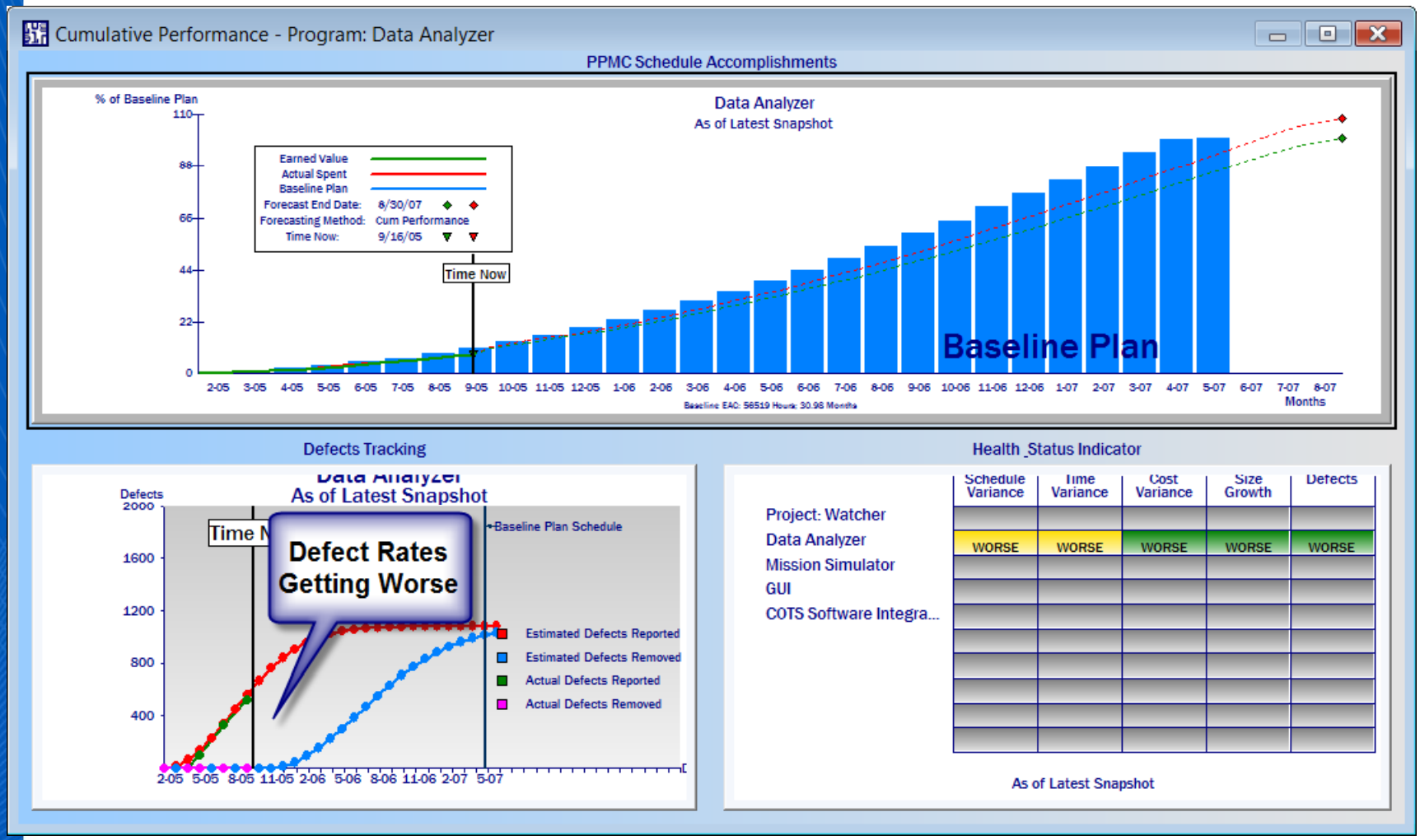
- Provide Measurement Results
- Project Planning
  - Establish Estimates
  - Develop a Project Plan
  - Obtain Commitment to the Plan
- Project Monitoring and Control
  - Monitor Project Against Plan
  - Manage Corrective Action to Closure
- Measurement and Analysis
  - Align Measurement and Analysis Activities



# What To Measure: Multiplicity of Metrics

1. Obvious: Status / Trend Metrics: e.g. productivity, defects removal rate, cost, schedule
2. Most important for improvement: Effectiveness ( 5 max)
   - "What we are doing that we should not do" e.g. number of delivered critical defects
   - "What we are not doing that we should do" e.g. number of defects that got past inspections
   - These metrics may change over time as we improve

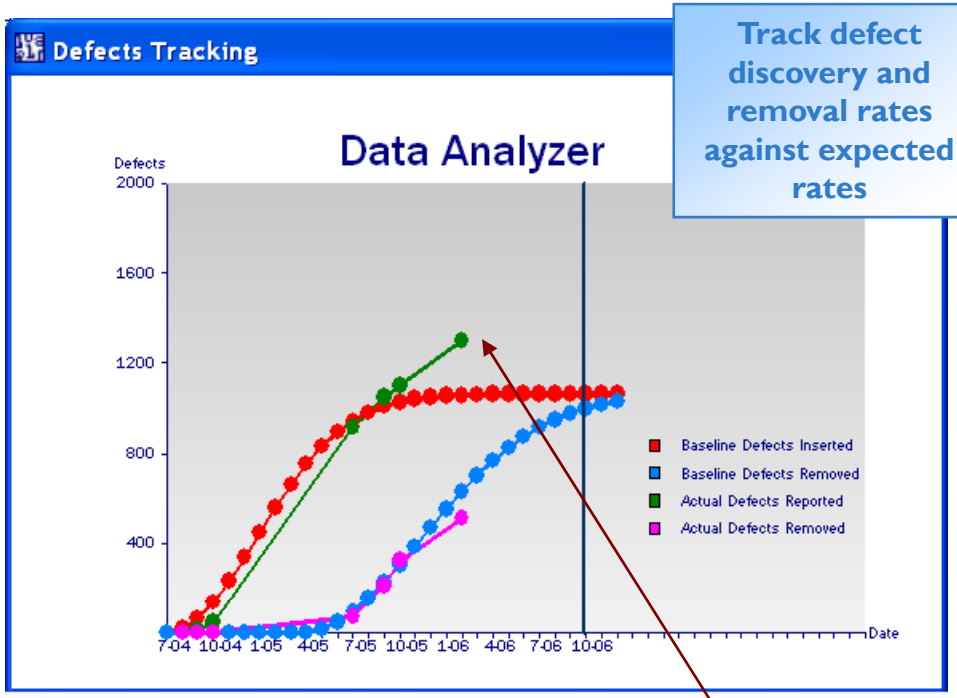
# Defects and Size Growth Provide Early Warning



# Defects and Growth Impact Software Process

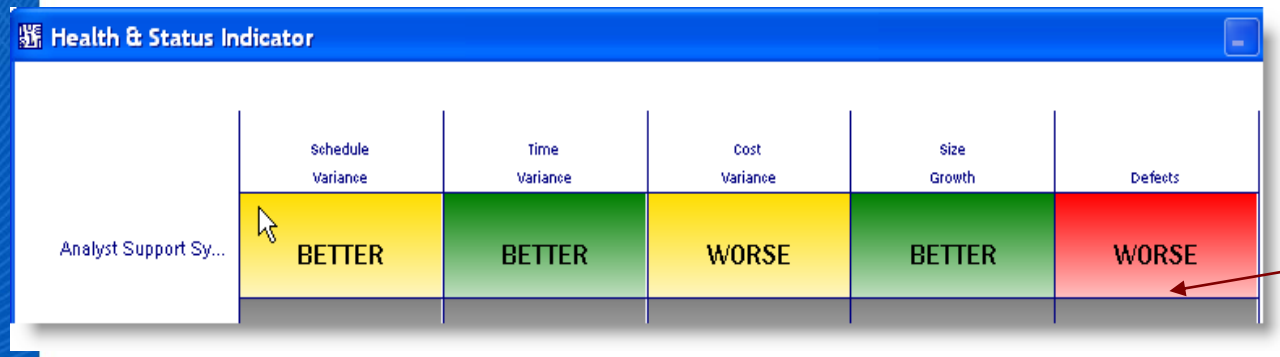
SEER by GALORATH

Track defect discovery and removal rates against expected rates




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

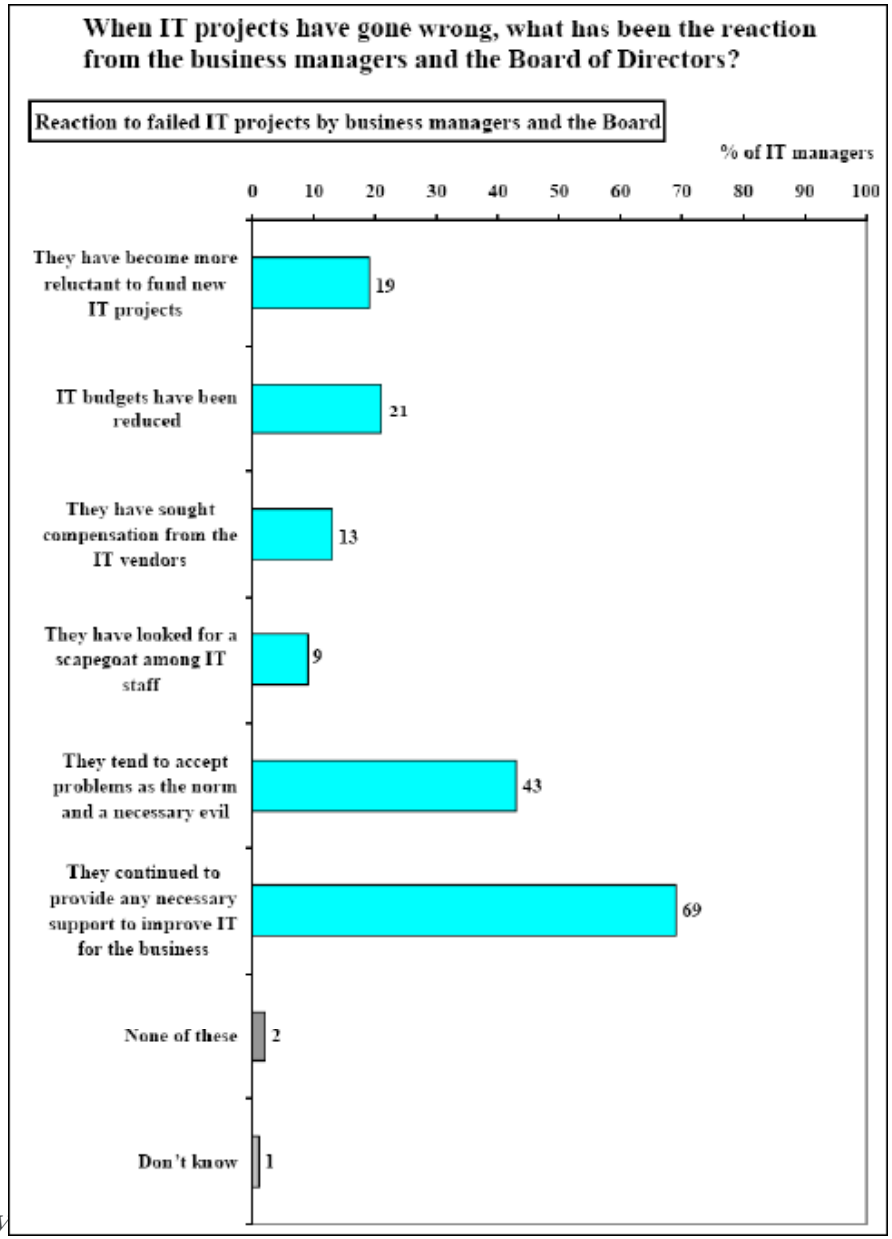
Thresholds are user definable




Increased defect reporting rate shows a worsening trend

# Measuring Defect Insertion & Removal During Development: Better Progress Measure Than Just Earned Value

# Tata Survey. When IT Projects Go Wrong



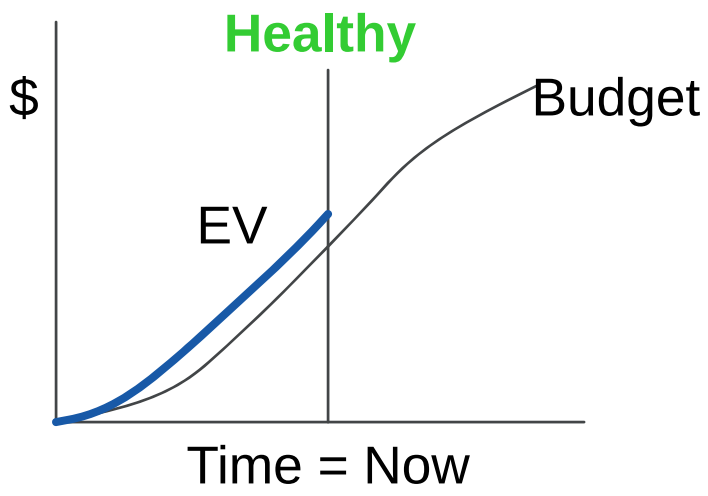
# Measure What Is Real

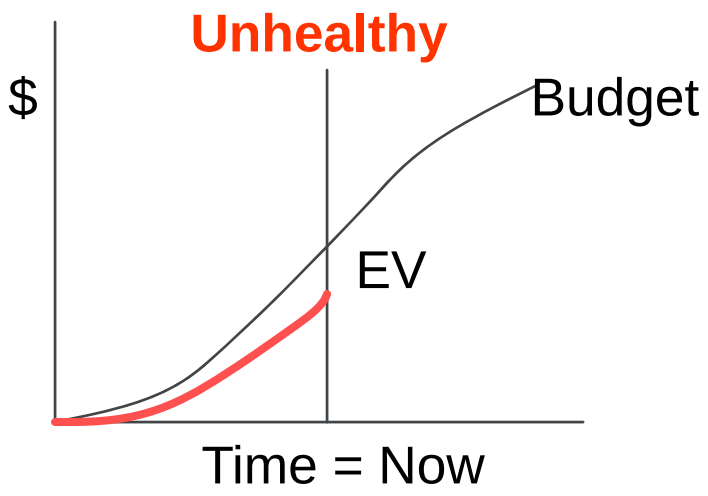
- "The government [is] extremely fond of amassing great quantities of statistics. These are raised to the nth degree, the cube roots are extracted, and the results are arranged into elaborate and impressive displays.
- What must be kept ever in mind, however, is that in every case, the figures are first put down by a village watchman, and he puts down anything he d..m well pleases.
- Attributed to Sir Josiah Stamp, 1840-1941, H.M. collector of inland revenu

How often are our measurement programs providing less than perfect data? E.g software cost data

# Use Earned Value TO Quantify Progress Versus Effort FOR DEVELOPMENT AND MAINTENANCE

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



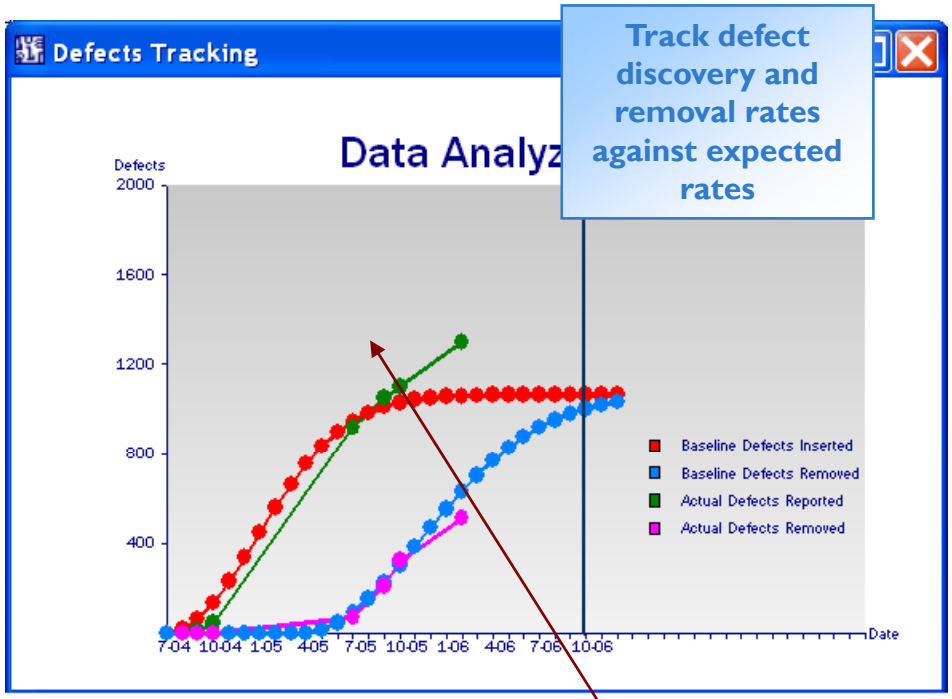



# Defects and Growth Impact Software Process

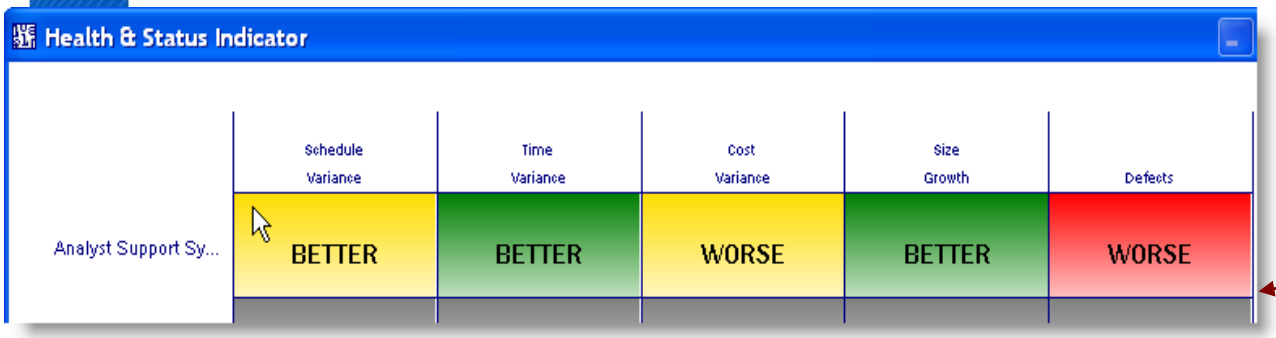
SEER by GALORATH

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

Thresholds are user definable



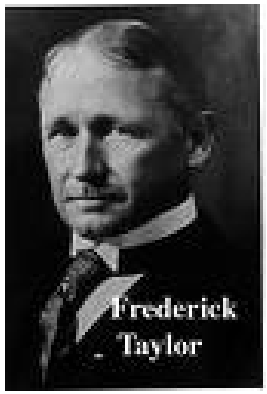

Track defect discovery and removal rates against expected rates




Increased defect reporting rate shows a worsening trend

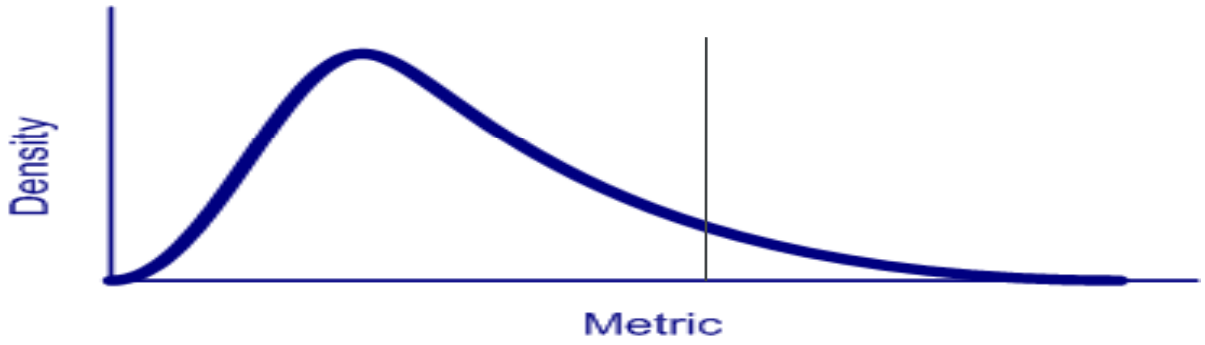
# The Hawthorne Effect: People Respond To Being Measured

- Goal: Find optimum for productivity 1924 to 1927
- **Increase, No Control Group**; Three departments; all showed an increase of productivity, whether illumination increased or decreased.
- **Increase, Control group** =change in lighting; experimental group got sequence of increasing light. Both groups substantially increased production, no difference between groups
- **Decrease, Control group got stable lights**; other sequence of decreasing levels. Both groups steadily increased production until the light in experimental group got so low they protested and production fell off
- All back to original: Productivity went up



# Manual Estimates: Human Reasons For Error (Metrics Can Help)

- Desire for “credibility” motivates overestimate behavior (80% probability?)
  - So must spend all the time to be “reliable”
  - Better approach force 50% probability & have “buffer” for overruns
- Technical pride causes underestimates
- Buy-in causes underestimates



# Core Metric: Value Provided By Software

- Concept: Spend where you obtain the most value
  - Value = savings to company or additional revenue due to the software
- Software Fails to add value much too often
  - Users enamored with concept
  - Concept deployed
  - Little to no value contributed to company…
  - Many reasons… often no changes in business rules
- MRP is a classic example of software hyped but which did not provide value

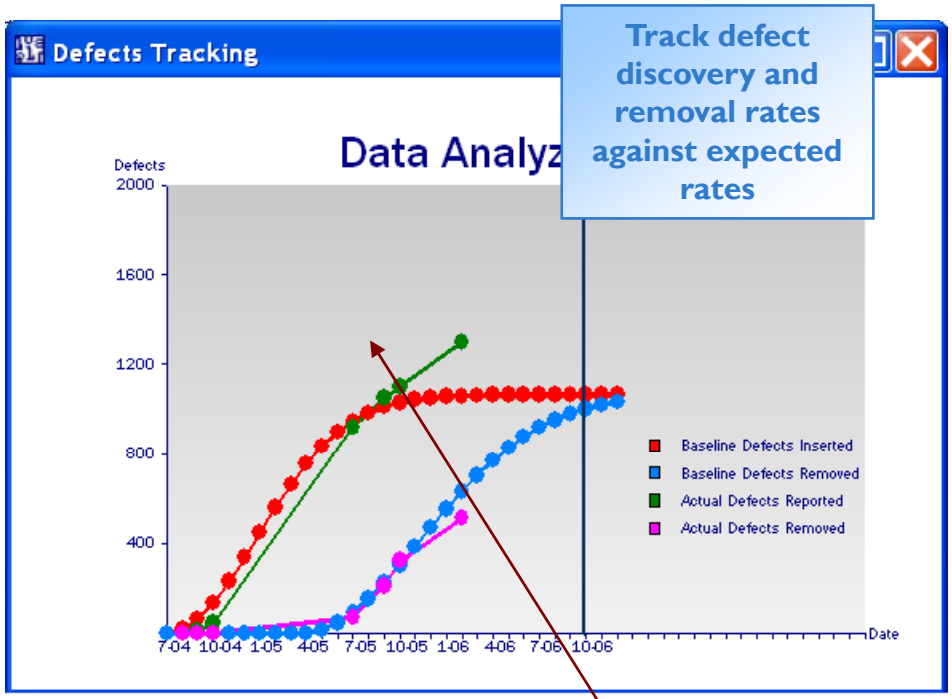
**Many Organizations May Not Be Mature Enough To Consider Value From the Software Team**

# Defects and Growth Impact Software Process

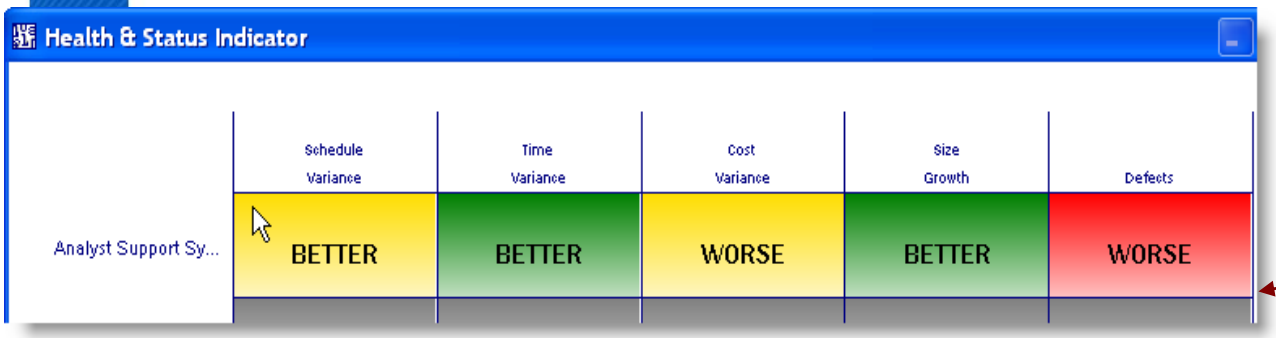
SEER by GALORATH

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

Thresholds are user definable

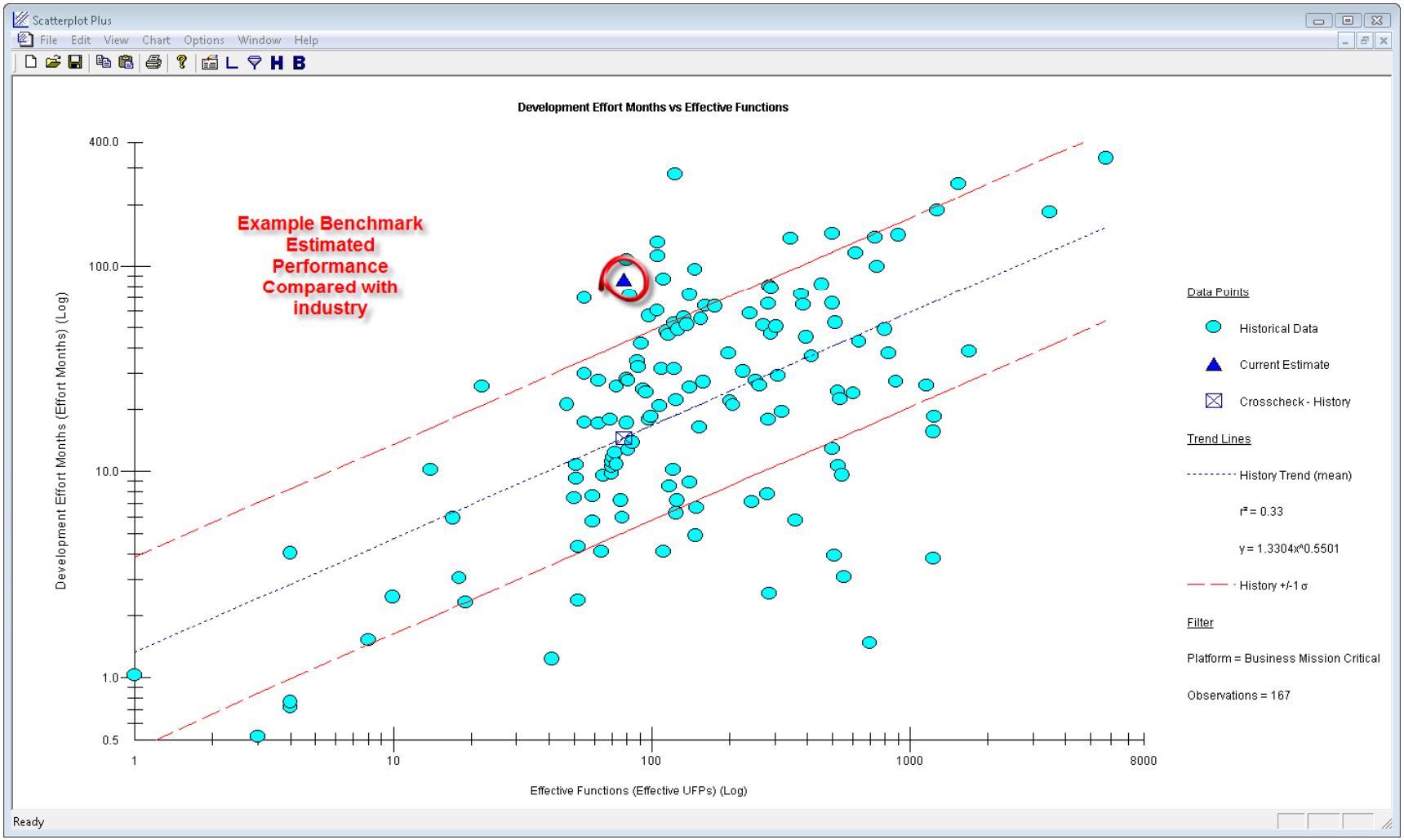



Track defect discovery and removal rates against expected rates



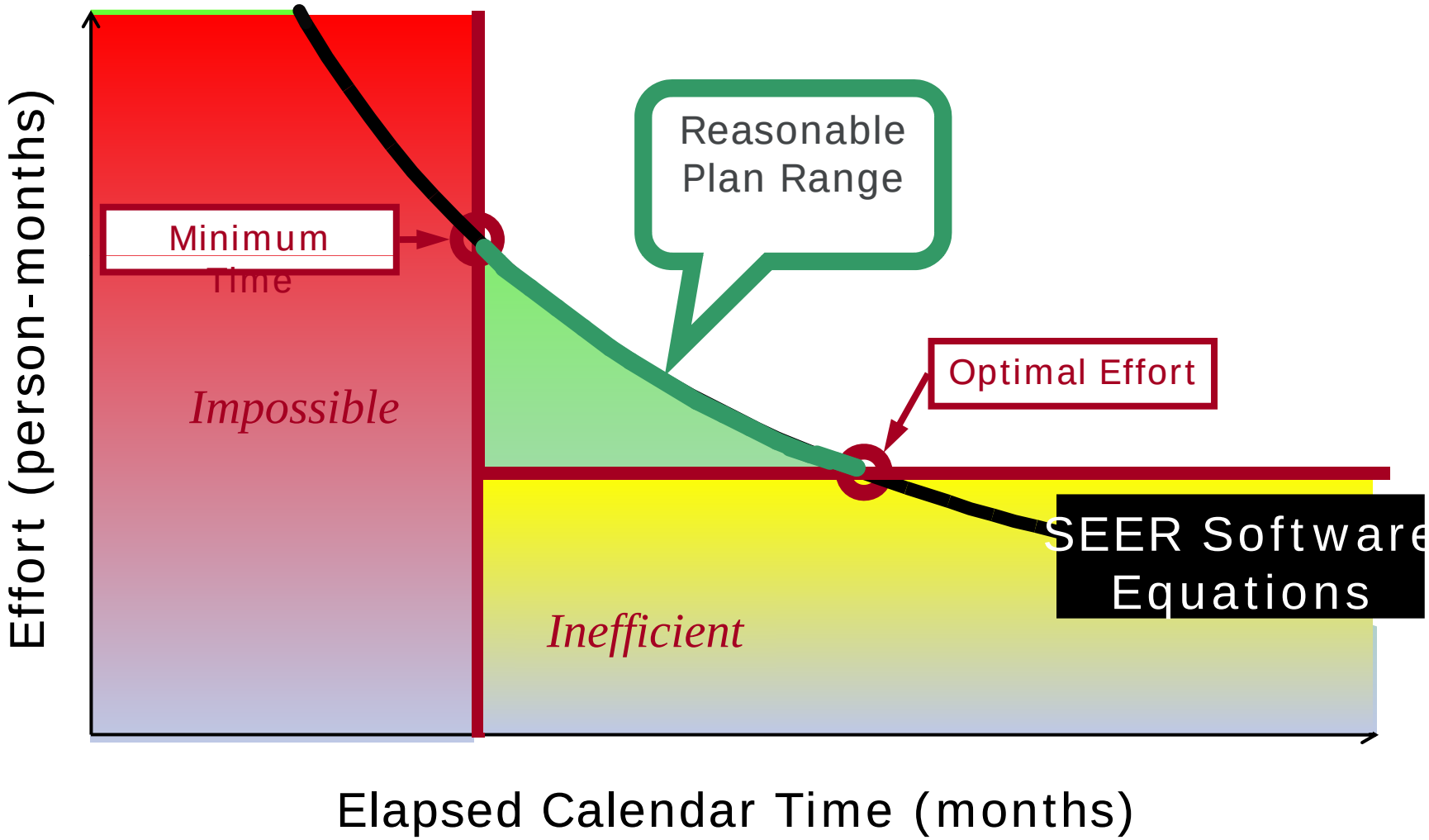

Increased defect reporting rate shows a worsening trend

# Example Benchmark Versus an Estimate.. Why Are We So Expensive?

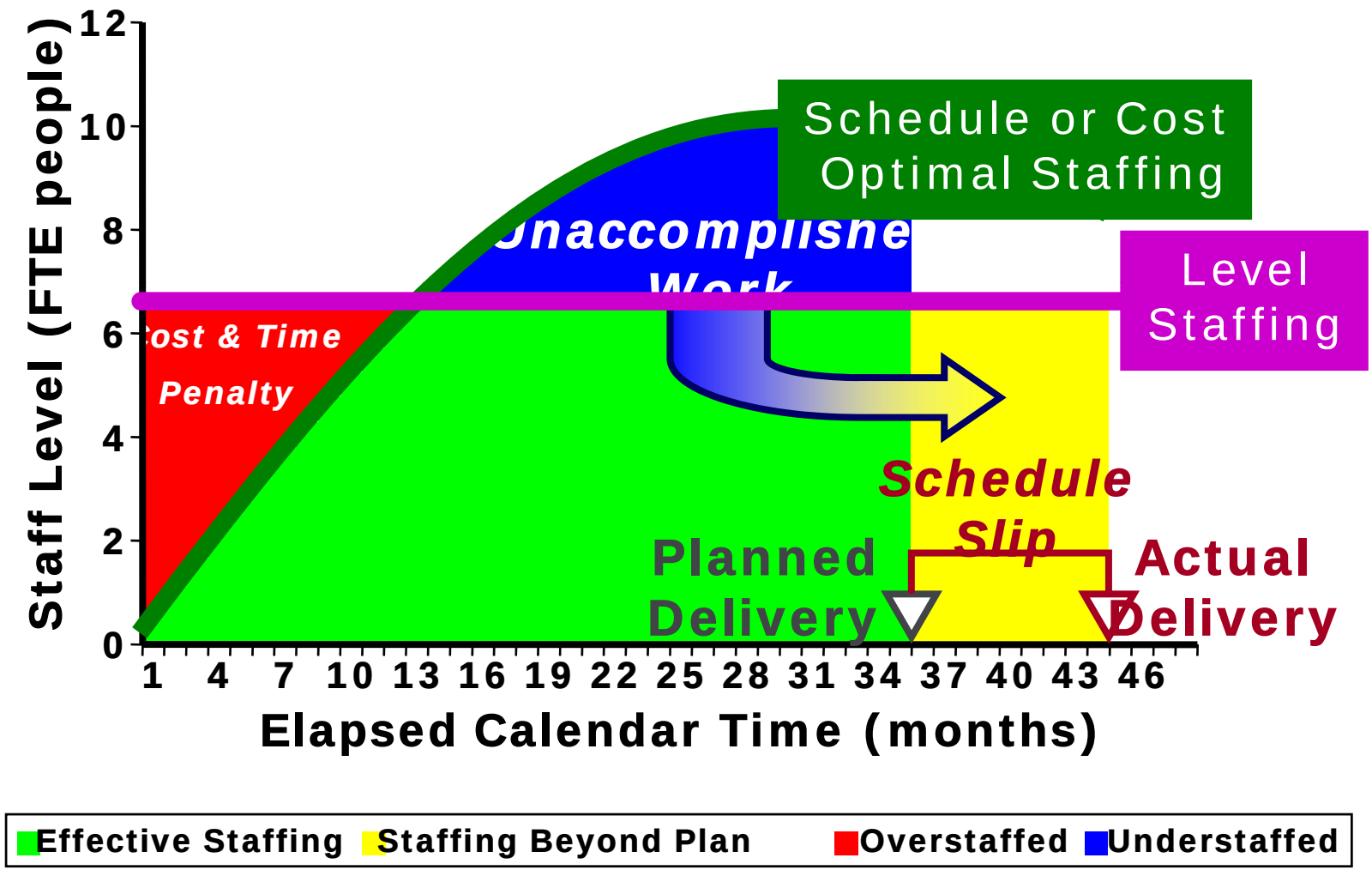


# Knowing Software Planning Possibilities Is Critical To Success

***For a given Size,Technology, Complexity & Probability***



# SEER-SEM: Avoid "Death Marches" and Failed Projects By Applying "Brooks Law"



estimate

estimate • analyze • plan • control

# Software Maintenance & Total Cost of Ownership

SEER by GALORATH

# Maintenance Defined

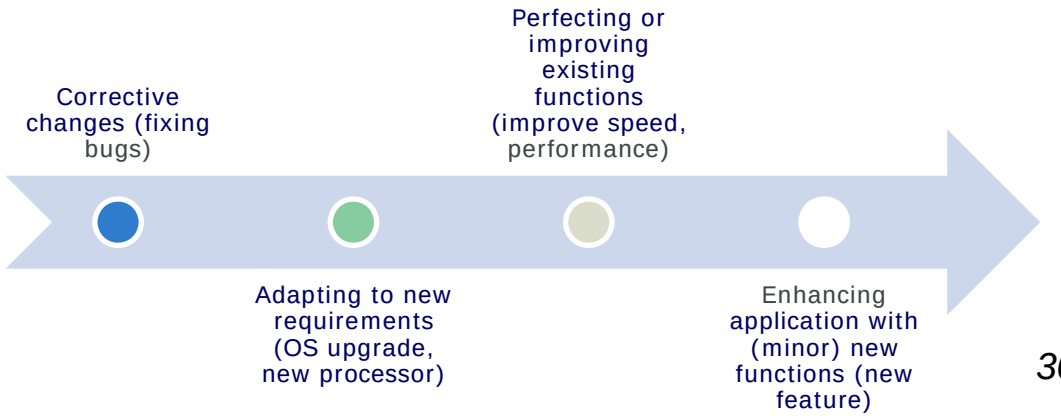
- Dictionary: "The work of keeping something in proper order"
- Software maintenance is different from hardware maintenance because:
  - Software doesn't physically wear out, but...
  - Software often gets less useful with age and...
  - It may be delivered with undiscovered flaws
- Software maintenance is: "The process of modifying existing operational software while leaving its primary functions intact."
- IEEE 1219, "The 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"

# Maintenance Standards Provide Definition and Consistency

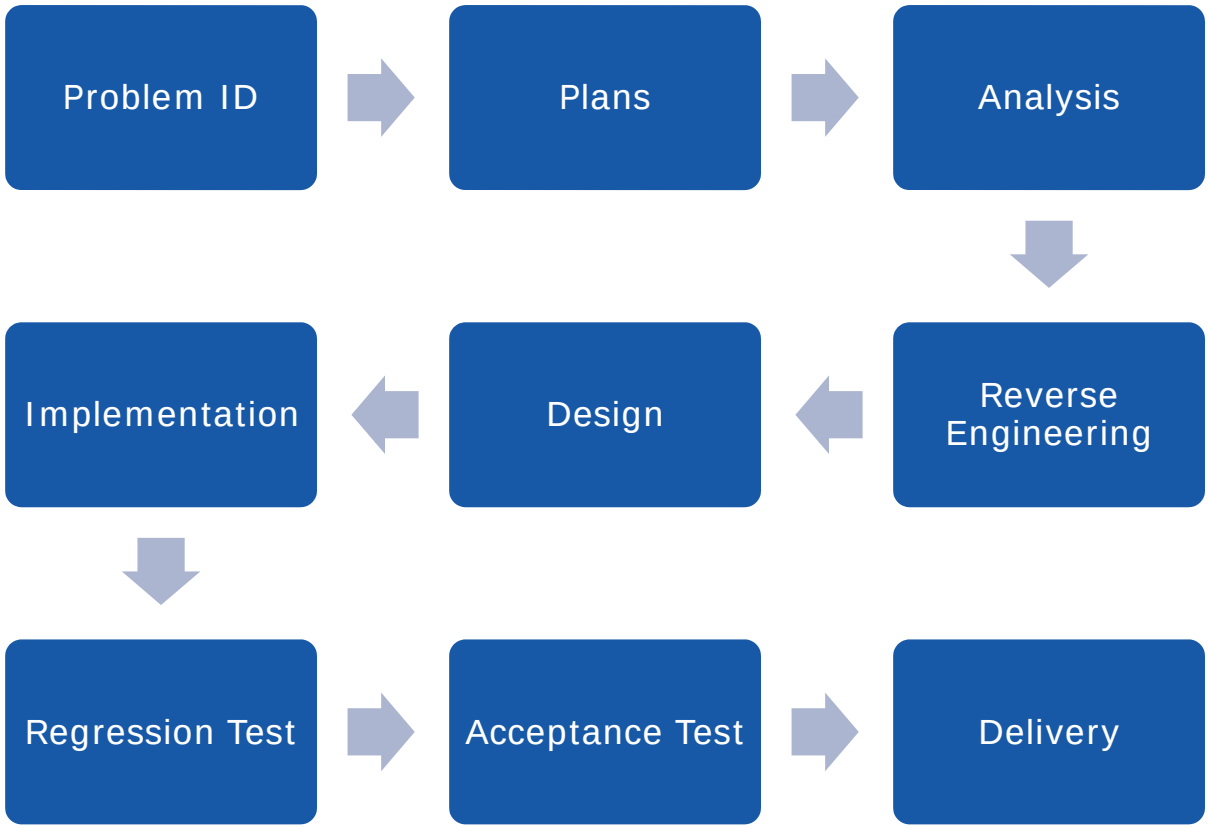
- IEEE 1219 Standard for Software Maintenance: Modification of software product after delivery to 1) correct faults, 2) improve performance or other attributes, 3) adapt the product to a modified environment
- IEEE/EIA 12207 standard for software life cycle processes: Process of a software product undergoing "modification to code and associated documentation due to a problem or the need for improvement"
  - The objective is to modify the existing software product while preserving its integrity."
- ISO/IEC 14764, international standard for software maintenance: defines software maintenance in the same terms as IEEE/EIA 12207
  - Emphasizes the pre-delivery aspects of maintenance, planning

# Maintenance Dissected

- Maintenance typically 75% + of the total software workload:
  - Highly dependent on rigor & operational "life expectancy"
  - Reducing maintenance costs can reduce life cycle costs <u>significantly</u>
- 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



# Maintenance Phases / Activities IEEE Et Al…




Many Development Metrics Are Applicable In Addition To Maintenance Metrics

# Software Maintenance Critical Success Factors (Source IEEE)

SEER by GALORATH

- 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

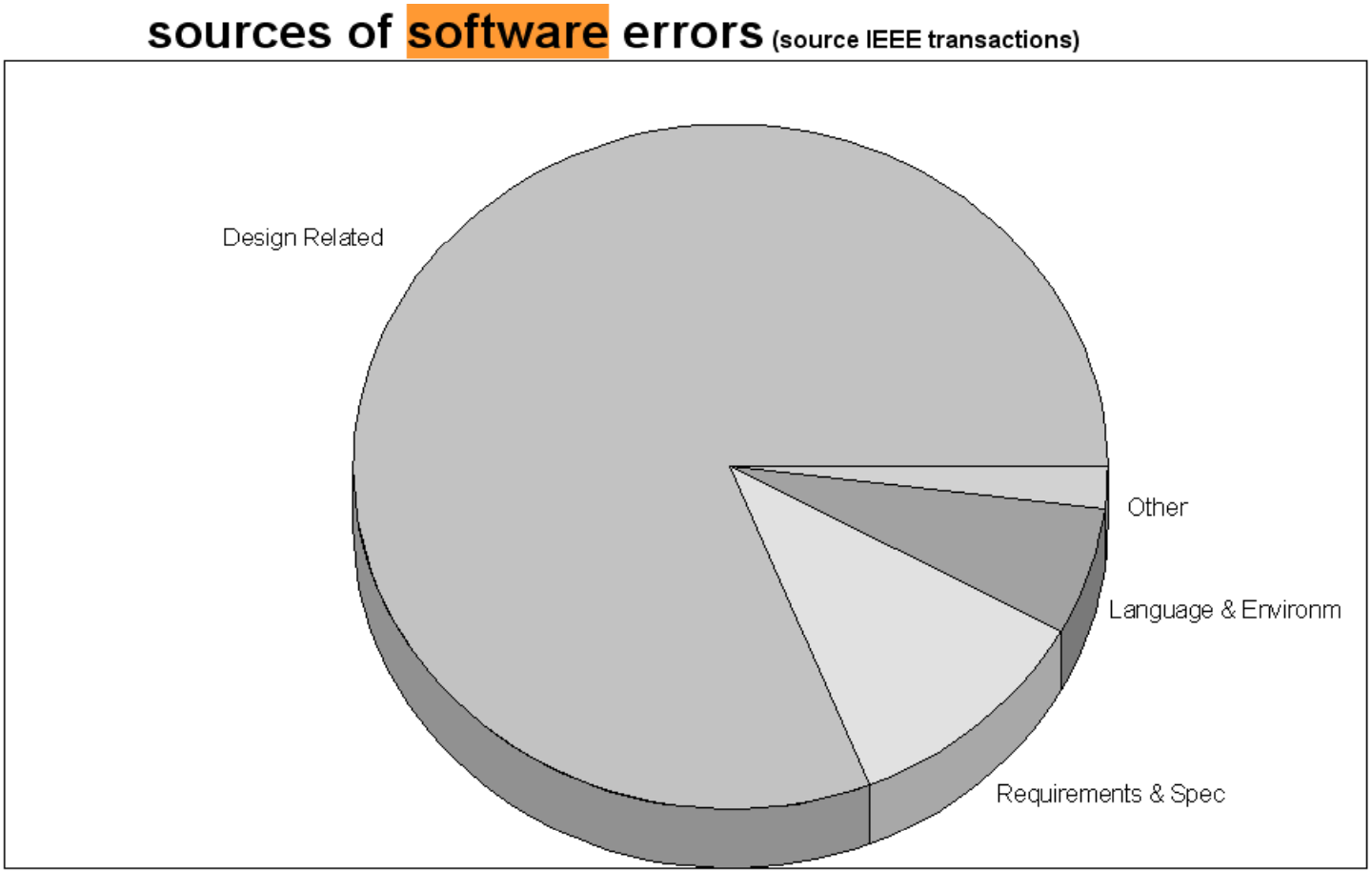
- May not have had maintenance as a goal
- System may not have been fully tested
- Documentation may be inadequate
- Maintenance staff may be inexperienced
- The tendency to produce quick & dirty fixes
- Process or language experience may have left a mess
- The "but I only changed 1 line syndrome"

# Why Software Maintenance Metrics Are 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

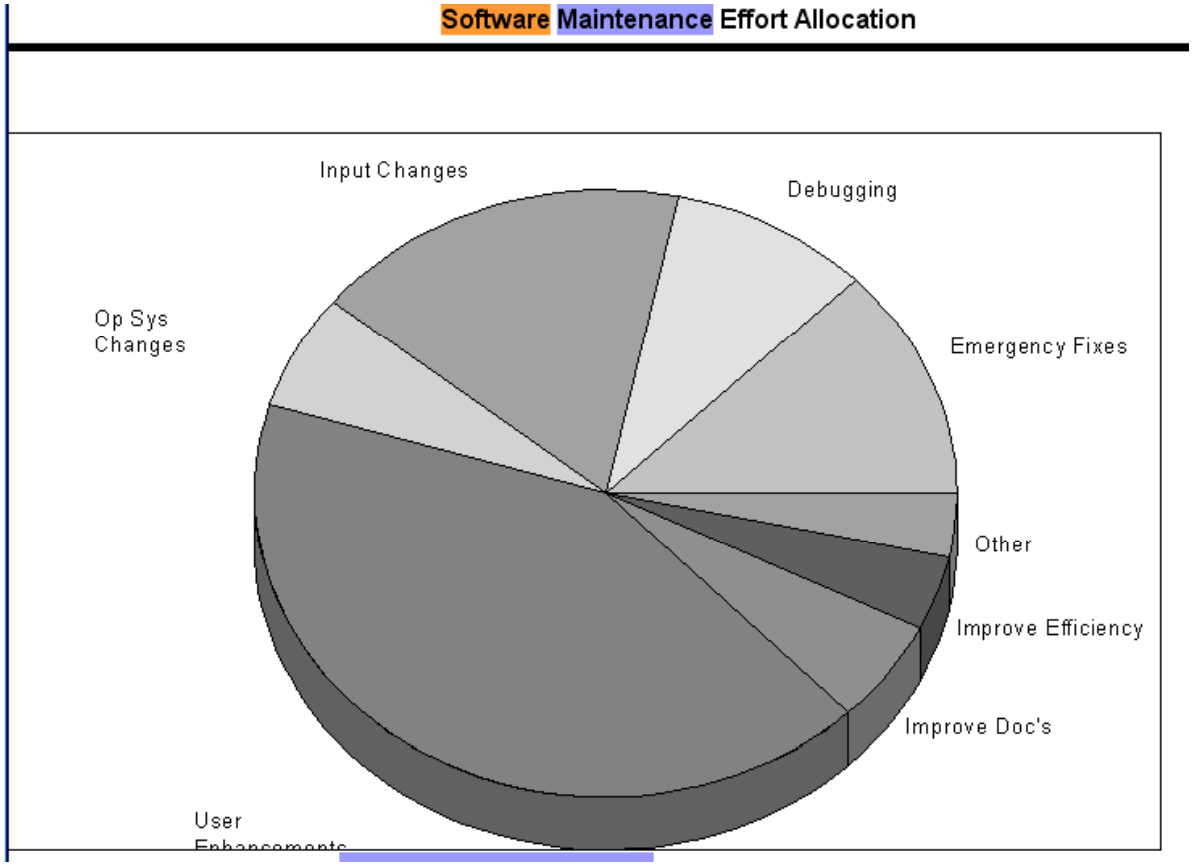
| Maintaining A Car | Maintaining Software |
|---|---|
| **High Maintenance**: Go By The Book (Regular Oil Changes, Etc.) | • Fix emergencies<br>• Provide new functionality as needed<br>• Adapt as necessary<br>• Software may not degenerate over time |
| **Nominal Maintenance**: Go Partially By The Book (Less Frequent Oil Changes, Etc. | • Fix emergencies<br>• Provide some required new functionality<br>• 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<br>• Software will degenerate over time |

# Sources of Software Errors

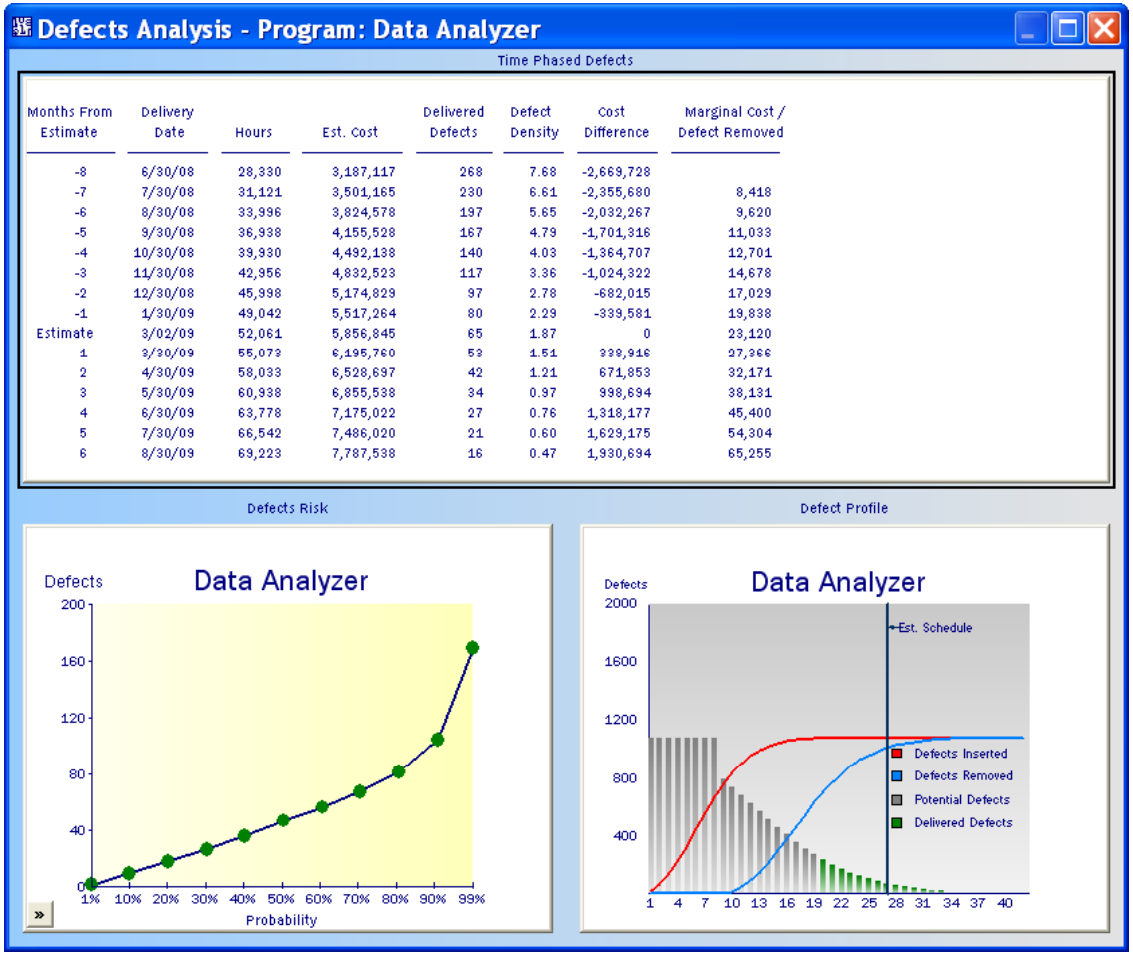



Software Maintenance Effort Allocation

# Allocation of Software Effort (Source IEEE)



# Development Defects Analysis Is a Clue to Maintenance Issues



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 | 55,079 | 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

# Measurement Job Not Over When Development Is Complete Maintenance GQM (Adapted from Mitre)

| Goal | Question | Metric(s) |
|---|---|---|
| Maximize Customer Satisfaction | How many problems affect the customer? | 1. Current Change Backlog<br>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<br>Software Maintainability<br>Computer resource Utilization |
| | Are we meeting delivery schedules? | Percentage of On-Time Deliveries |

# Example Maintenance Metrics

- Defects Inserted per correction
- Defects removed per unit time
- Productivity for block changes
- Maintainability
- Mean time to find the next k faults
- Maintenance backlog
- Increases / decrease on maintenance backlog
- Number of trouble reports opened and closed

# More Example Maintenance Metrics

- Mean time until problem closed
- Defects during warranty period
- Mean time to resolution
- Defects by type and severity
- Time to respond to customer reported defects
- **Mccabe & Halstead complexity metrics**

# Software Maturity Index (Example of Metric from IEEE 982 Standard Dictionary of Measures to Produce Reliable Software)

M = number of modules in current version

A = number of added modules in current version

C = number of changed modules in current version

D = number of deleted modules in current version compared to the previous version

SMI = (M - (A + C + D)) / M

- when SMI approaches 1.0 the product is stable

# Example Effectiveness metrics for Maintenance

- Number of new defects created by fixes
- Number of defect corrections that were not correct
- Number of defects not repaired in promised time (Delinquent)
- **Defect Seepage.. (Customer reported defects during pre-delivery testing)**

Identify the metrics that YOUR organization needs

# Product Age / Technology Metrics

- Becomes increasingly difficult to maintain older technology
- Would you recommend a student study COBOL, Ada or PASCAL

- People become less available
- Tools an practices become obsolete

# Maintenance Productivity Drivers: Scope

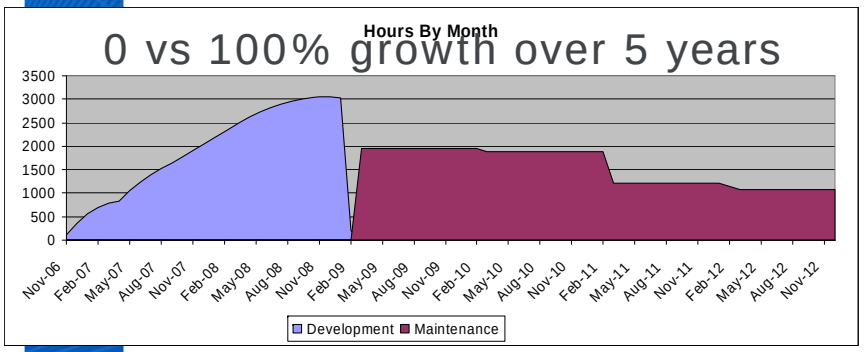
- Years of Maintenance
  - Number of years for which software maintenance costs will be estimated
    - Maintenance typically begins when operational test & evaluation is completed
- Separate Sites
  - Number of separate operational sites where the software will be installed and users will have an input 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

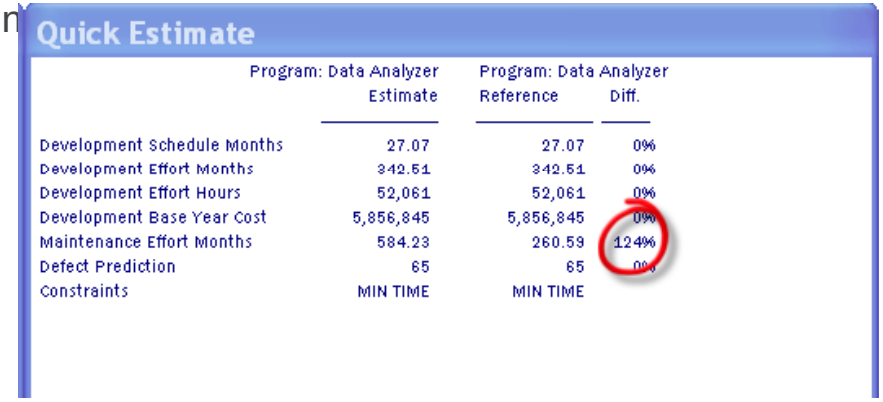
# Maintenance Growth Over Life

- 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 enhan |
| 0% | Sustaining engineering only |

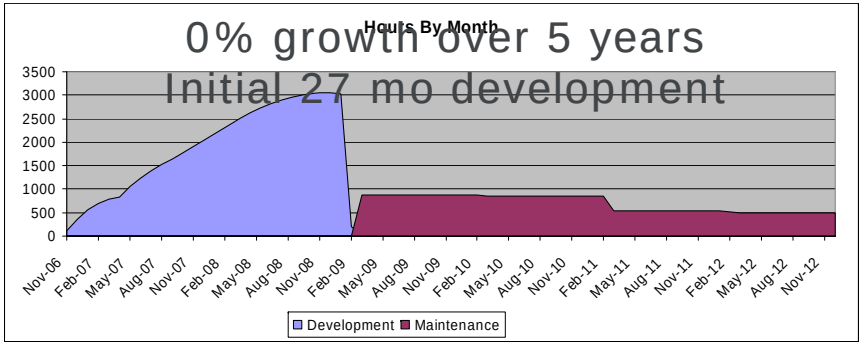
100% growth over 5 years
Initial 27 mo development



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 | |



# 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 |

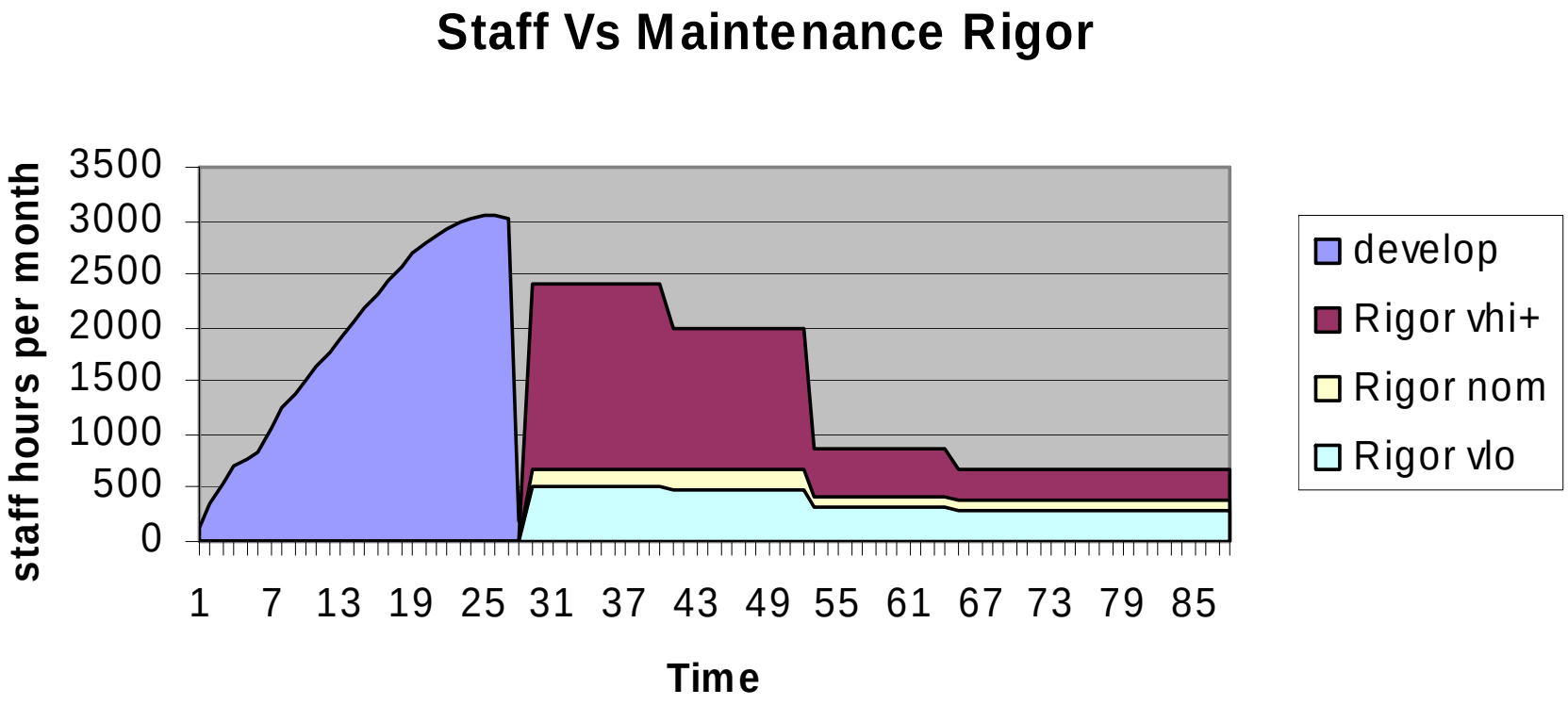
50% vs 0 annual change over 5 years

**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)

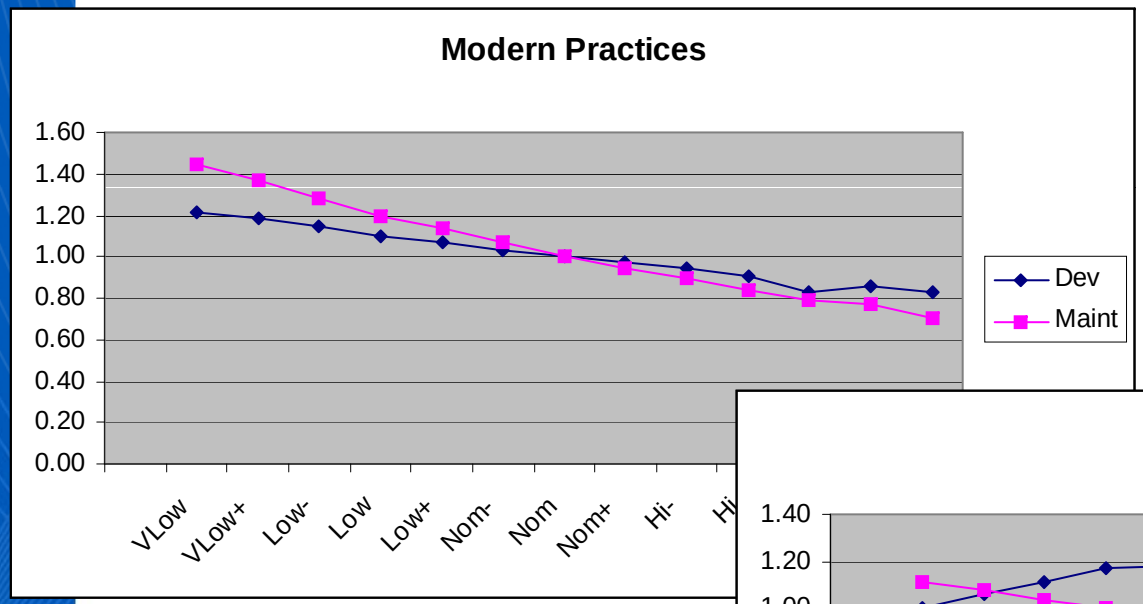
## Most Projects Spend Low During Maintenance

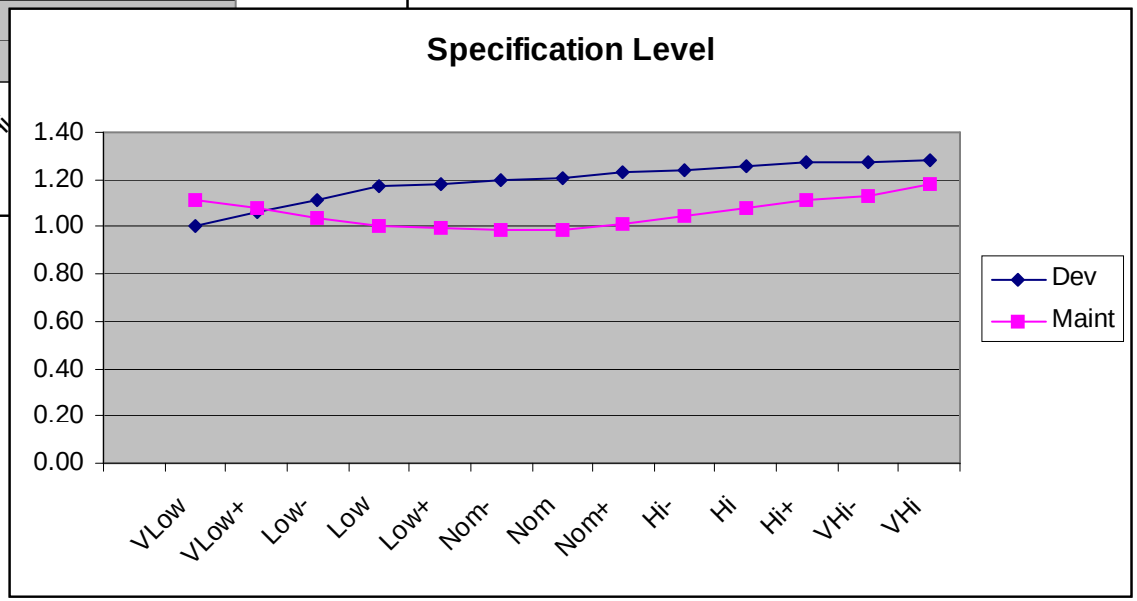


# Percent to be Maintained

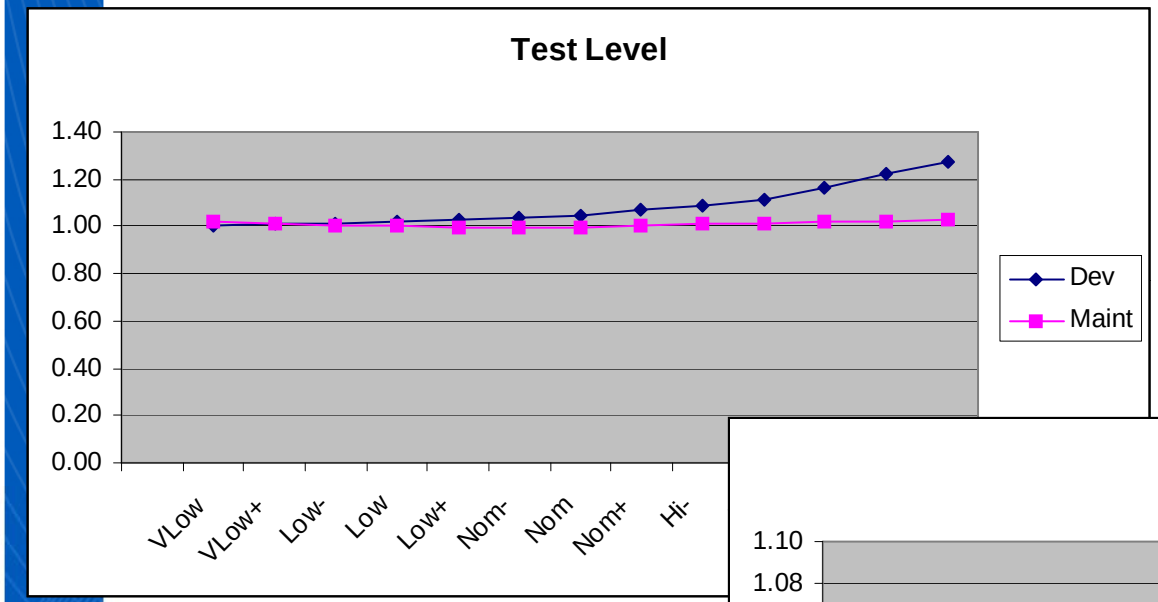
- Percent of the total code that will be maintained
- If maintenance will be shared with another organization, only the portion to be included in this estimate
- If software cannot be changed, exclude it from the percent to be maintained (e.g. non updateable embedded processors)

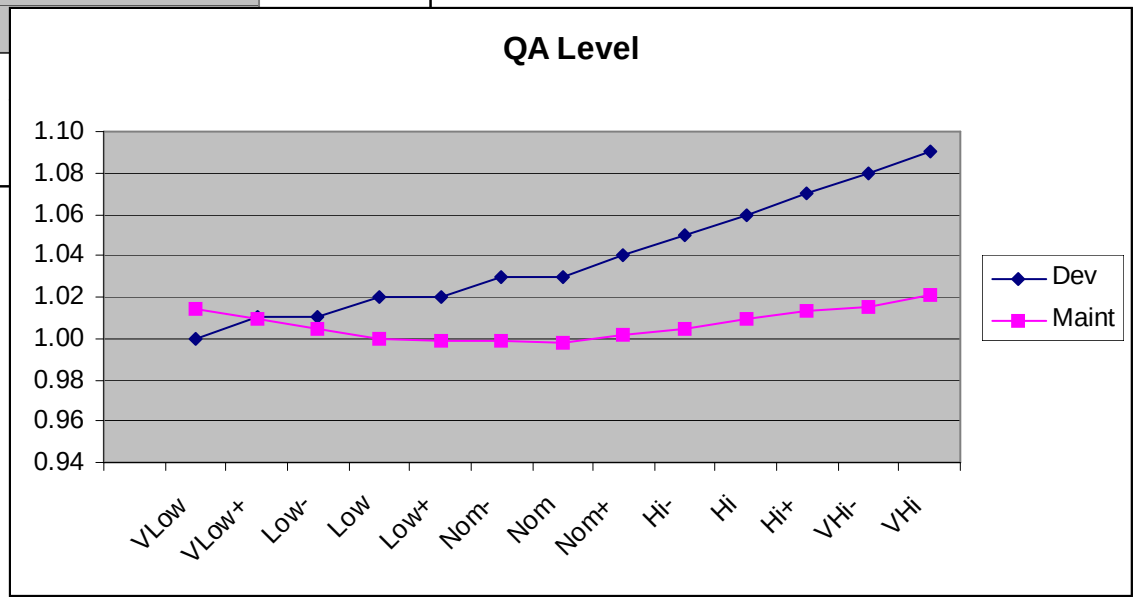
# People, Process, Technology Sensitivity Development Vs Maintenance – 1



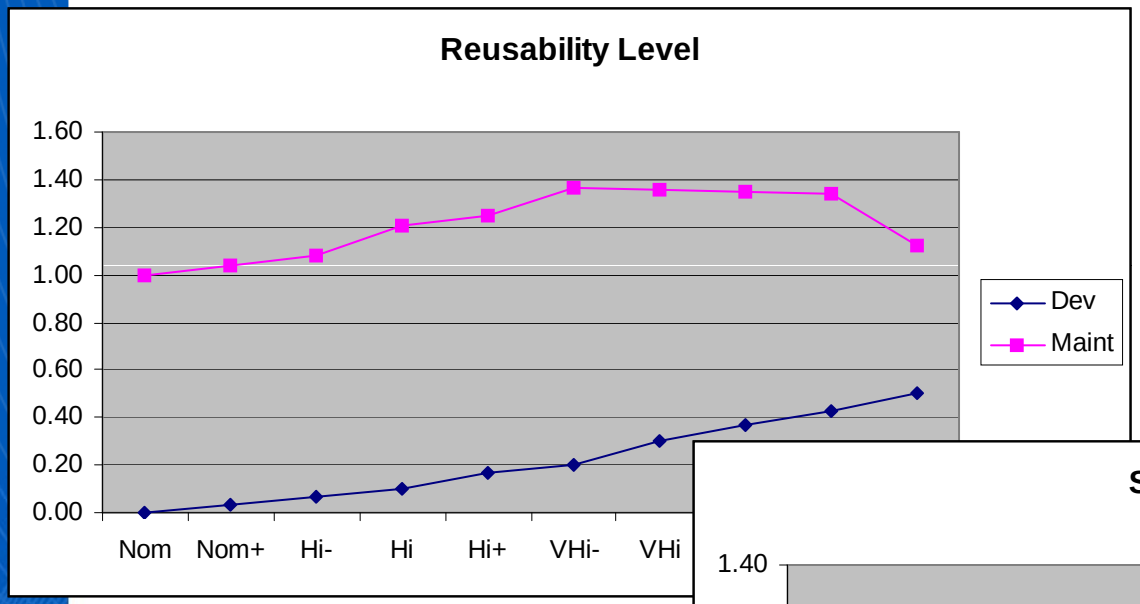



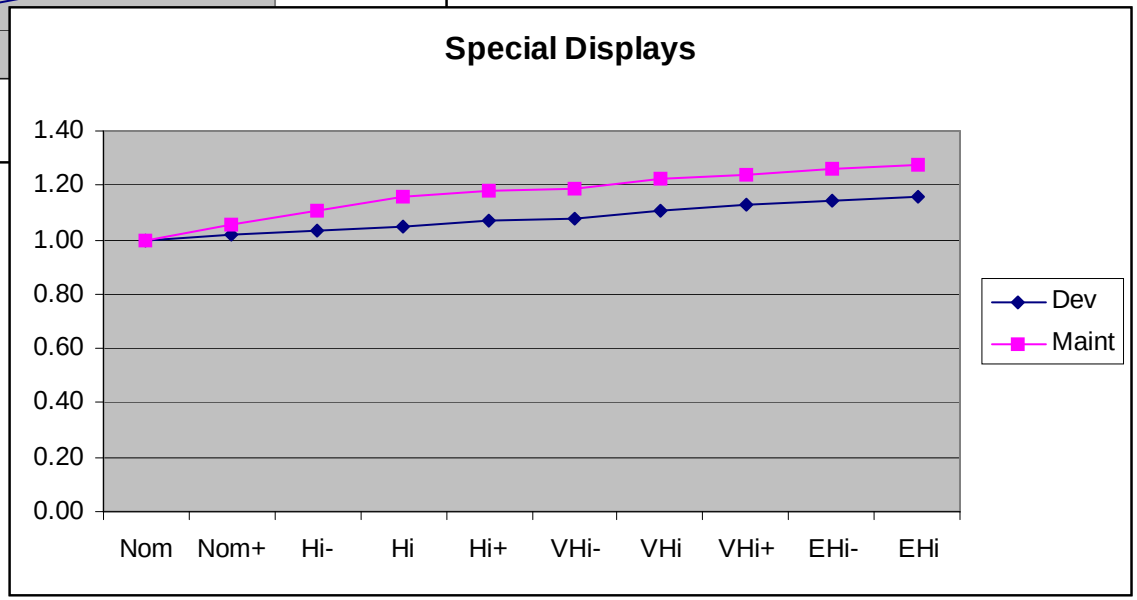
# Development Vs Maintenance - 2



# Development Vs Maintenance - 3



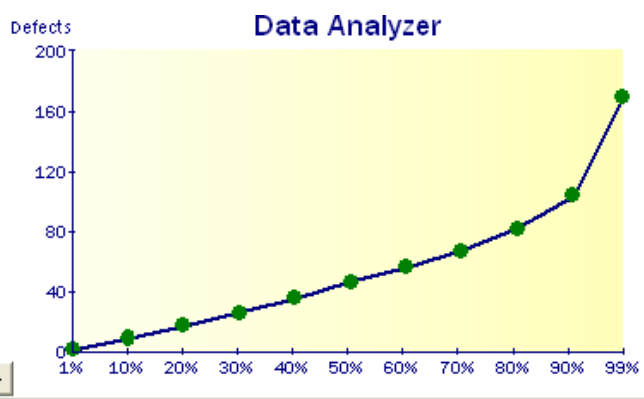
# Defects Can Be Reduced By Further Development Testing but Not Eliminated

Defects Analysis - Program: Data Analyzer

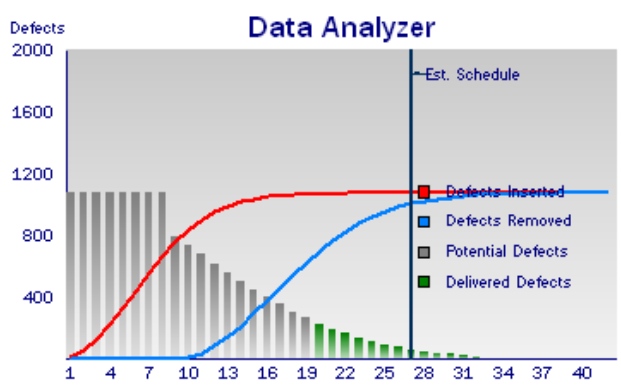
Time Phased Defects

| 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



# 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 | |

# 7 Characteristics of a Dysfunctional Software Projects (Source: Mike Evans, et al.)

- Based on 350 projects:
- 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

- Maintenance can be 75% of total ownership costs
- Development decisions, processes and tools impact maintenance costs
- While software maintenance is often treated as a level of effort activity there are consequences:
  - Quality, functionality and reliability
- Consider total ownership costs and risks
- Applied measurement is a critical component of software and systems management
- Measure what you want people to focus on
- Continue emphasis on standards and definition

# SEER by Galorath Offerings

**Projects Delivered...**

On Time. On Budget.
As Specified.

- Products:

SEER for Software

SEER for IT

SEER for Hardware, Electronics & Systems

SEER for Manufacturing


SEER Services & Support

Dan Galorath
310 414-3222 x614
galorath@galorath.com
Blog: www.galorath.com/wp