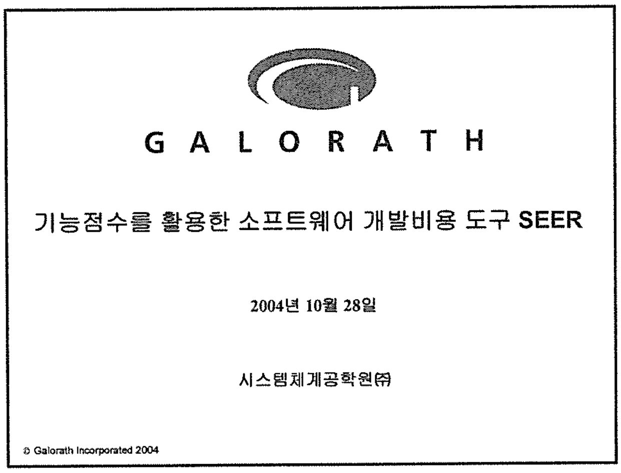
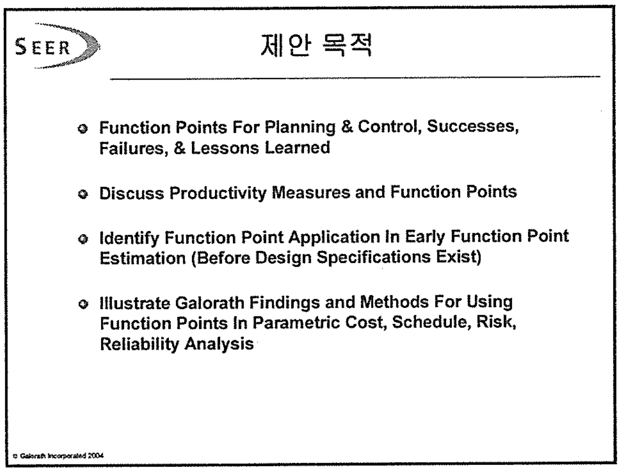


G A L O R A T H

기능점수를 활용한 소프트웨어 개발비용 도구 SEER

2004년 10월 28일

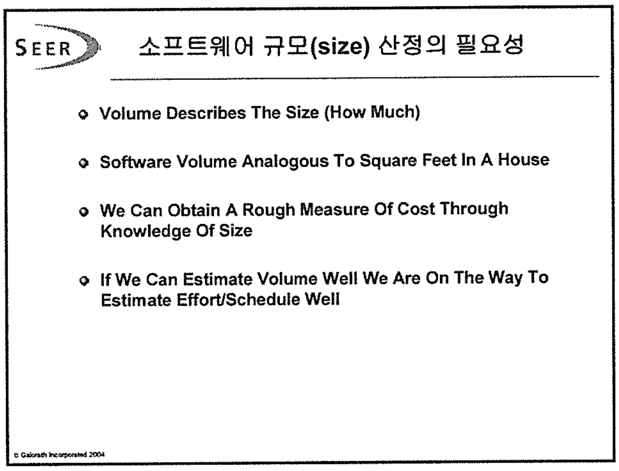
시스템체계공학원㈜



SEER

제안 목적

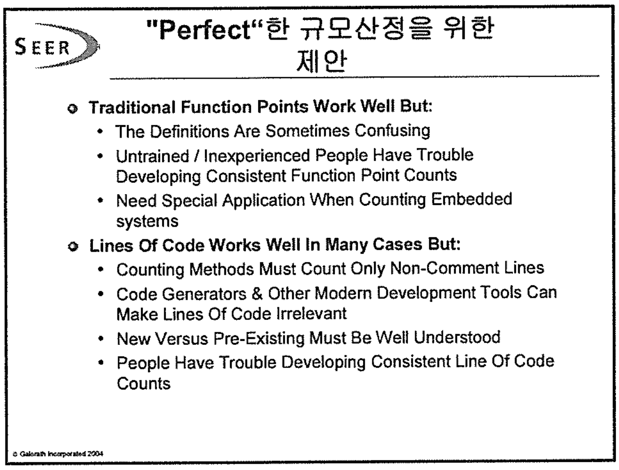
- Function Points For Planning & Control, Successes, Failures, & Lessons Learned
- Discuss Productivity Measures and Function Points
- Identify Function Point Application In Early Function Point Estimation (Before Design Specifications Exist)
- Illustrate Galorath Findings and Methods For Using Function Points In Parametric Cost, Schedule, Risk, Reliability Analysis



SEER

소프트웨어 규모(size) 산정의 필요성

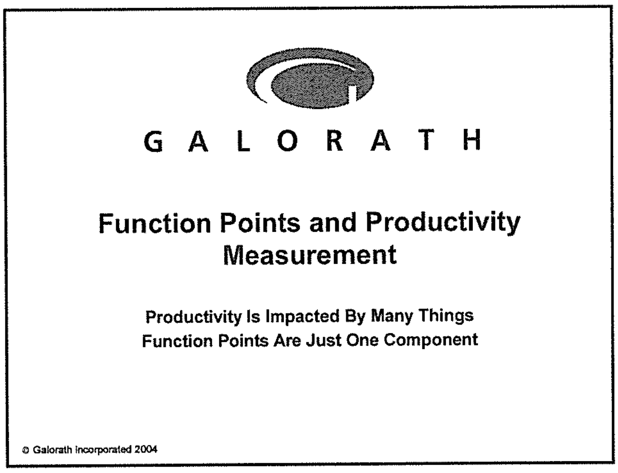
- Volume Describes The Size (How Much)
- Software Volume Analogous To Square Feet In A House
- We Can Obtain A Rough Measure Of Cost Through Knowledge Of Size
- If We Can Estimate Volume Well We Are On The Way To Estimate Effort/Schedule Well



SEER

"Perfect"한 규모산정을 위한 제안

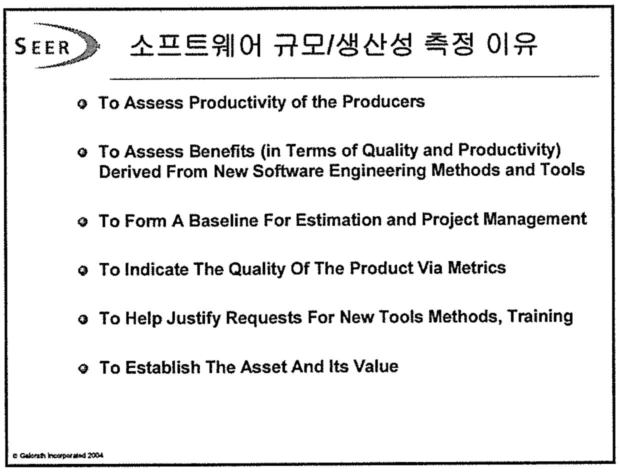
- Traditional Function Points Work Well But:
  - The Definitions Are Sometimes Confusing
  - Untrained / Inexperienced People Have Trouble Developing Consistent Function Point Counts
  - Need Special Application When Counting Embedded systems
- Lines Of Code Works Well In Many Cases But:
  - Counting Methods Must Count Only Non-Comment Lines
  - Code Generators & Other Modern Development Tools Can Make Lines Of Code Irrelevant
  - New Versus Pre-Existing Must Be Well Understood
  - People Have Trouble Developing Consistent Line Of Code Counts



G A L O R A T H

Function Points and Productivity Measurement

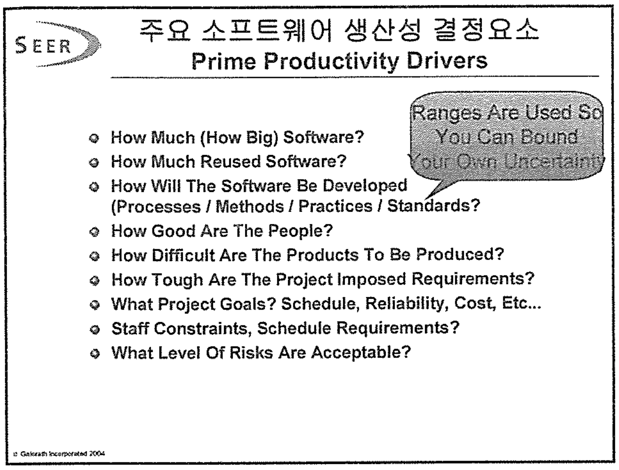
Productivity Is Impacted By Many Things
Function Points Are Just One Component



SEER

소프트웨어 규모/생산성 측정 이유

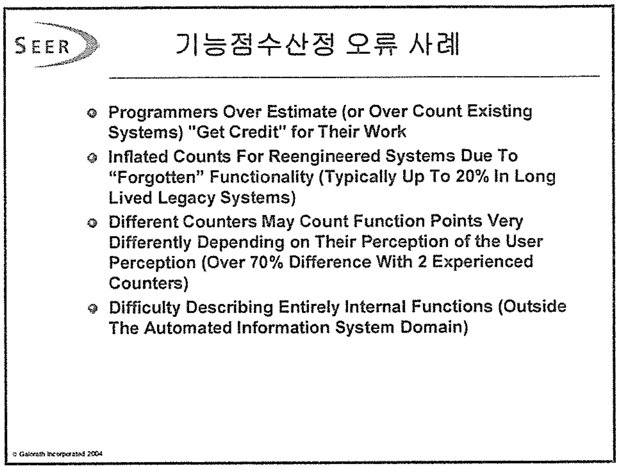
- To Assess Productivity of the Producers
- To Assess Benefits (in Terms of Quality and Productivity) Derived From New Software Engineering Methods and Tools
- To Form A Baseline For Estimation and Project Management
- To Indicate The Quality Of The Product Via Metrics
- To Help Justify Requests For New Tools Methods, Training
- To Establish The Asset And Its Value



주요 소프트웨어 생산성 결정요소
Prime Productivity Drivers

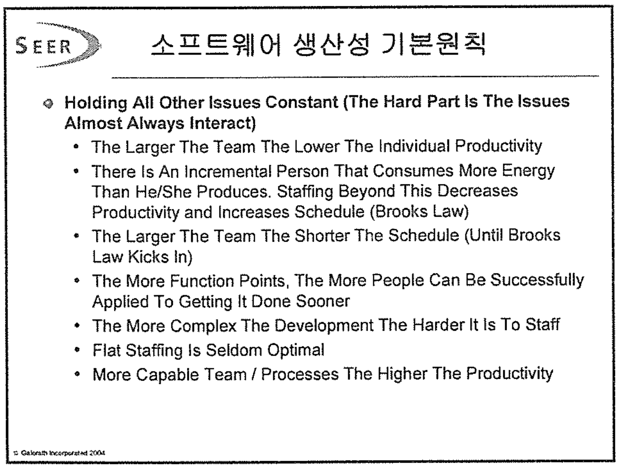
- How Much (How Big) Software?
- How Much Reused Software?
- How Will The Software Be Developed (Processes / Methods / Practices / Standards?
- How Good Are The People?
- How Difficult Are The Products To Be Produced?
- How Tough Are The Project Imposed Requirements?
- What Project Goals? Schedule, Reliability, Cost, Etc...
- Staff Constraints, Schedule Requirements?
- What Level Of Risks Are Acceptable?

Ranges Are Used So You Can Bound Your Own Uncertainty



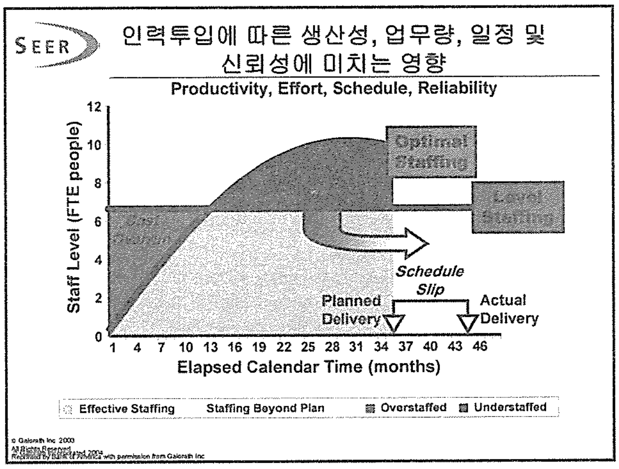
기능점수산정 오류 사례

- Programmers Over Estimate (or Over Count Existing Systems) "Get Credit" for Their Work
- Inflated Counts For Reengineered Systems Due To "Forgotten" Functionality (Typically Up To 20% In Long Lived Legacy Systems)
- Different Counters May Count Function Points Very Differently Depending on Their Perception of the User Perception (Over 70% Difference With 2 Experienced Counters)
- Difficulty Describing Entirely Internal Functions (Outside The Automated Information System Domain)



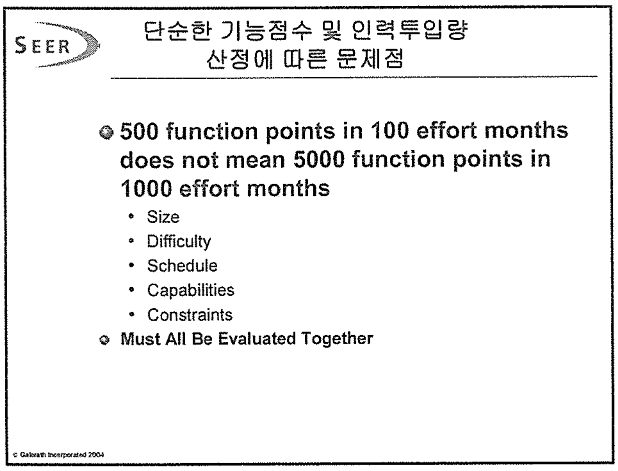
소프트웨어 생산성 기본원칙

- Holding All Other Issues Constant (The Hard Part Is The Issues Almost Always Interact)
  - The Larger The Team The Lower The Individual Productivity
  - There Is An Incremental Person That Consumes More Energy Than He/She Produces. Staffing Beyond This Decreases Productivity and Increases Schedule (Brooks Law)
  - The Larger The Team The Shorter The Schedule (Until Brooks Law Kicks In)
  - The More Function Points, The More People Can Be Successfully Applied To Getting It Done Sooner
  - The More Complex The Development The Harder It Is To Staff
  - Flat Staffing Is Seldom Optimal
  - More Capable Team / Processes The Higher The Productivity



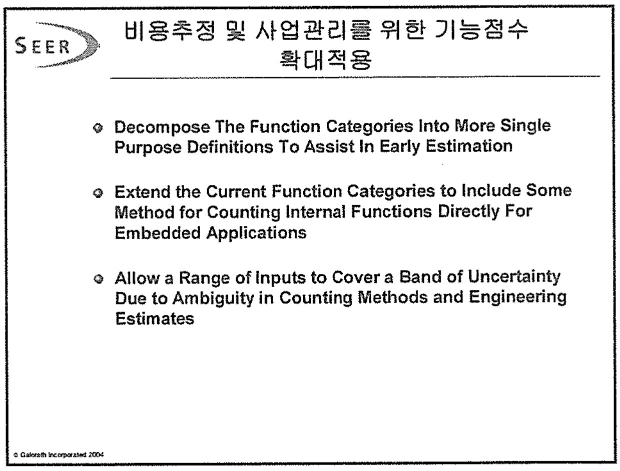
인력투입에 따른 생산성, 업무량, 일정 및 신뢰성에 미치는 영향

Productivity, Effort, Schedule, Reliability



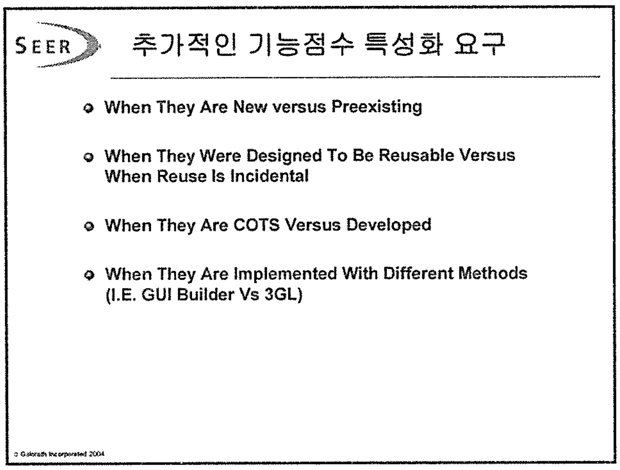
단순한 기능점수 및 인력투입량 산정에 따른 문제점

- 500 function points in 100 effort months does not mean 5000 function points in 1000 effort months
  - Size
  - Difficulty
  - Schedule
  - Capabilities
  - Constraints
- Must All Be Evaluated Together



비용추정 및 사업관리를 위한 기능점수 확대적용

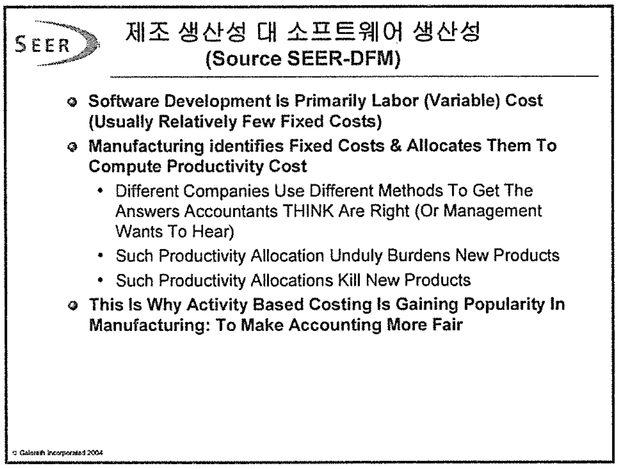
- Decompose The Function Categories Into More Single Purpose Definitions To Assist In Early Estimation
- Extend the Current Function Categories to Include Some Method for Counting Internal Functions Directly For Embedded Applications
- Allow a Range of Inputs to Cover a Band of Uncertainty Due to Ambiguity in Counting Methods and Engineering Estimates



추가적인 기능점수 특성화 요구

- When They Are New versus Preexisting
- When They Were Designed To Be Reusable Versus When Reuse Is Incidental
- When They Are COTS Versus Developed
- When They Are Implemented With Different Methods (I.E. GUI Builder Vs 3GL)

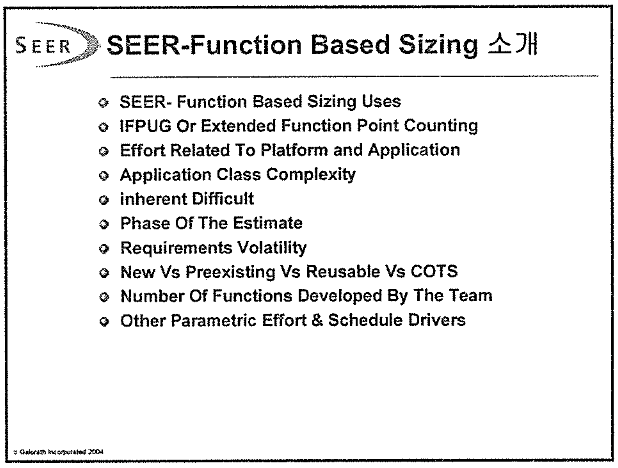
© Galorath Incorporated 2004



제조 생산성 대 소프트웨어 생산성 (Source SEER-DFM)

- Software Development Is Primarily Labor (Variable) Cost (Usually Relatively Few Fixed Costs)
- Manufacturing identifies Fixed Costs & Allocates Them To Compute Productivity Cost
  - Different Companies Use Different Methods To Get The Answers Accountants THINK Are Right (Or Management Wants To Hear)
  - Such Productivity Allocation Unduly Burdens New Products
  - Such Productivity Allocations Kill New Products
- This Is Why Activity Based Costing Is Gaining Popularity In Manufacturing: To Make Accounting More Fair

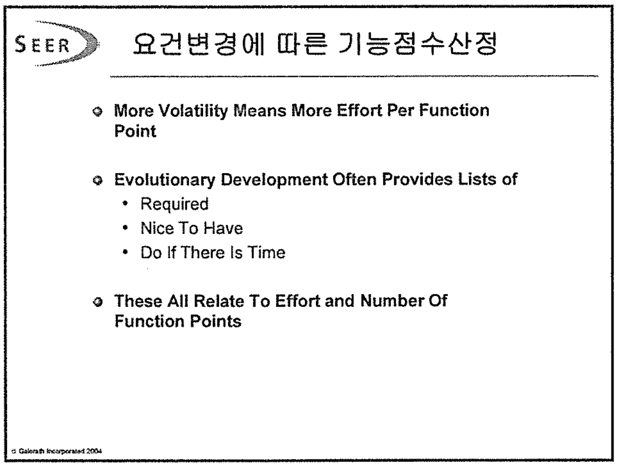
© Galorath Incorporated 2004



SEER-Function Based Sizing 소개

- SEER- Function Based Sizing Uses
- IFPUG Or Extended Function Point Counting
- Effort Related To Platform and Application
- Application Class Complexity
- inherent Difficult
- Phase Of The Estimate
- Requirements Volatility
- New Vs Preexisting Vs Reusable Vs COTS
- Number Of Functions Developed By The Team
- Other Parametric Effort & Schedule Drivers

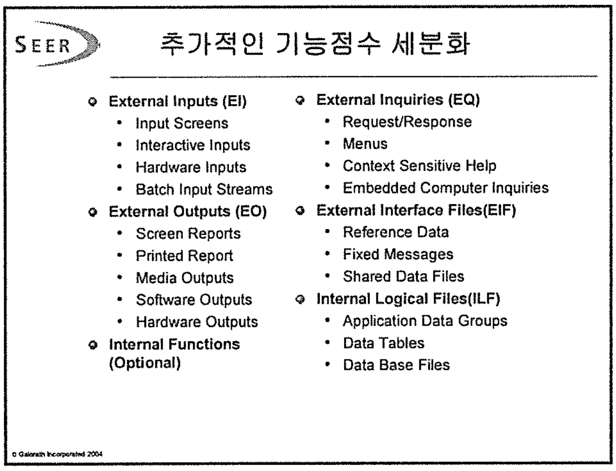
© Galorath Incorporated 2004



요건변경에 따른 기능점수산정

- More Volatility Means More Effort Per Function Point
- Evolutionary Development Often Provides Lists of
  - Required
  - Nice To Have
  - Do If There Is Time
- These All Relate To Effort and Number Of Function Points

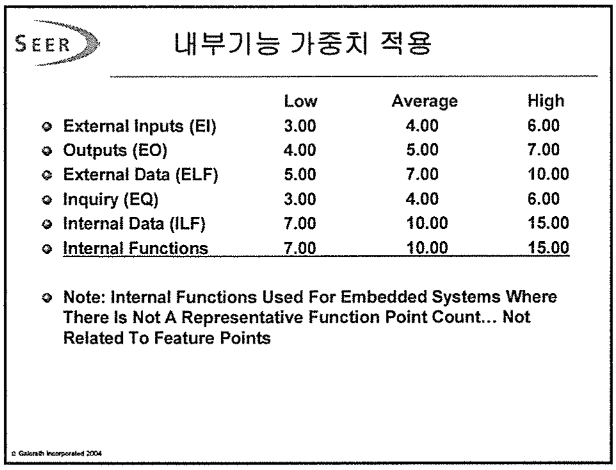
© Galorath Incorporated 2004



추가적인 기능점수 세분화

- External Inputs (EI)
  - Input Screens
  - Interactive Inputs
  - Hardware Inputs
  - Batch Input Streams
- External Outputs (EO)
  - Screen Reports
  - Printed Report
  - Media Outputs
  - Software Outputs
  - Hardware Outputs
- Internal Functions (Optional)
- External Inquiries (EQ)
  - Request/Response
  - Menus
  - Context Sensitive Help
  - Embedded Computer Inquiries
- External Interface Files(EIF)
  - Reference Data
  - Fixed Messages
  - Shared Data Files
- Internal Logical Files(ILF)
  - Application Data Groups
  - Data Tables
  - Data Base Files

© Galorath Incorporated 2004



내부기능 가중치 적용

	Low	Average	High
External Inputs (EI)	3.00	4.00	6.00
Outputs (EO)	4.00	5.00	7.00
External Data (ELF)	5.00	7.00	10.00
Inquiry (EQ)	3.00	4.00	6.00
Internal Data (ILF)	7.00	10.00	15.00
Internal Functions	7.00	10.00	15.00

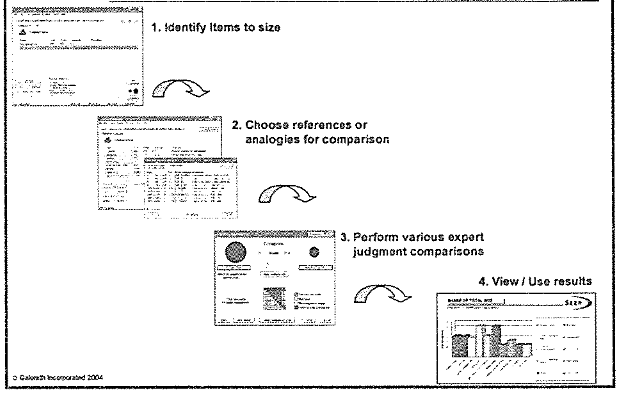
- Note: Internal Functions Used For Embedded Systems Where There Is Not A Representative Function Point Count... Not Related To Feature Points

© Galorath Incorporated 2004

AccuScope 적용 효과

- AccuScope allows users to…
- Size early – provide reasonable size ranges prior to a detailed understanding of project
- Size accurately - applying relative knowledge often results in more accurate up front estimates vs. other methods
- Estimate the number of function points for one or many systems without taking the time to count them

Four Step Sizing Process



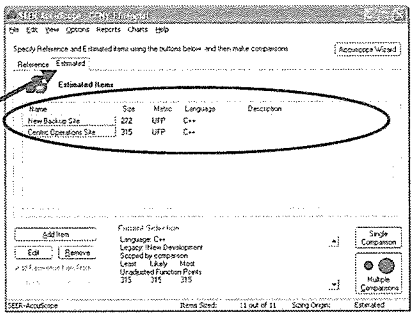
1. Identify Items to size

2. Choose references or analogies for comparison

3. Perform various expert judgment comparisons

4. View / Use results

Step 1: Identify Items To Size



Estimated items are those which will be sized.

Specify Reference and Estimated items using the buttons below and then make comparisons

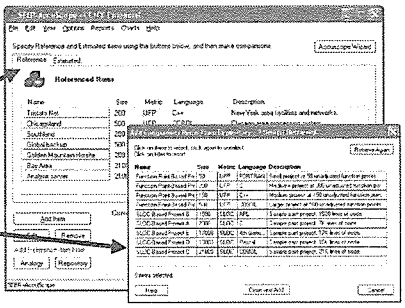
Estimated Items

Name	Size	Metric	Language	Description
New Backup Site	272	UFP	C++	
Centric Operations Site	315	UFP	C++	

Current Selection
Language: C++
Legacy: New Development
Scoped by comparison
Least Likely Most
Unadjusted Function Points
315 315 315

SEER-AccuScope Items Sized: 11 out of 11 Sizing Origin: Estimated

Step 2: Choose References To Size



Reference items are those from which comparisons are made.

These items can be entered manually or obtained from a repository.

Specify Reference and Estimated items using the buttons below, and then make comparisons.

Referenced Items

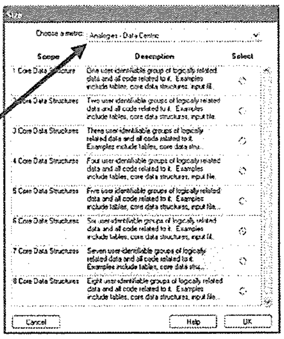
Name	Size	Metric	Language	Description
Tristate Net	200	UFP	C++	New York area facilities and networks.
Chicagoland	500	UFP	COBOL	[illegible]
Southland	200			
Global backup	500			
Golden Mountain Hotsite	200			
Bay Area	210			
Analysis server	210			

Click on items to select, click again to unselect
Click on titles to resort

Name	Size	Metric	Language	Description
Function Point Based Pr	50	UFP	FORTRAN	Small project at 50 unadjusted function points
Function Point Based Pr	200	UFP	C	Medium project at 200 unadjusted function po
Function Point Based Pr	150	UFP	C++	Medium project at 150 unadjusted function poin
Function Point Based Pr	500	UFP	JOVIAL	Large project at 500 unadjusted function points
SLOC Based Project B	1500	SLOC	APL	Sample past project, 1500 lines of code
SLOC Based Project A	7000	SLOC	C	Sample past project, 7K lines of code
SLOC Based Project E	17000	SLOC	4th Gene…	Sample past project, 17K lines of code
SLOC Based Project D	10000	SLOC	Pascal	Sample past project, 10K lines of code
SLOC Based Project C	21000	SLOC	COBOL	Sample past project, 21K lines of code

0 items selected

Step 2: Choose an Analogy



Analogies are optional.

Analogies are reference items based on past patterns.

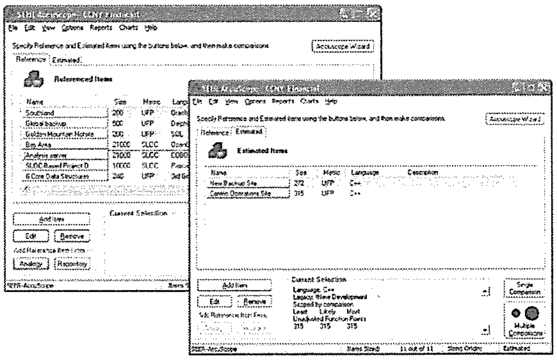
Choose from various sets of analogies, each representing different sizing perspectives, or create your own manually.

Size

Choose a metric: Analogies - Data Centric

Scope	Description	Select
1 Core Data Structure	One user-identifiable group of logically related data and all code related to it. Examples include tables, core data structures, input fil…	○
2 Core Data Structures	Two user-identifiable groups of logically related data and all code related to it. Examples include tables, core data structures, input file.	○
3 Core Data Structures	Three user-identifiable groups of logically related data and all code related to it. Examples include tables, core data stru…	○
4 Core Data Structures	Four user-identifiable groups of logically related data and all code related to it. Examples include tables, core data structures, input file.	○
5 Core Data Structures	Five user-identifiable groups of logically related data and all code related to it. Examples include tables, core data structures, input file…	○
6 Core Data Structures	Six user-identifiable groups of logically related data and all code related to it. Examples include tables, core data structures, input fil…	○
7 Core Data Structures	Seven user-identifiable groups of logically related data and all code related to it. Examples include tables, core data stru…	○
8 Core Data Structures	Eight user-identifiable groups of logically related data and all code related to it. Examples include tables, core data structures, input file…	○

Step 3: Ready For Comparisons



Specify Reference and Estimated items using the buttons below, and then make comparisons.

Referenced Items

Name	Size	Metric	Language
Southland	200	UFP	Oracle
Global backup	500	UFP	Delphi
Golden Mountain Hotsite	200	UFP	SQL
Bay Area	21000	SLOC	OpenE…
Analysis server	21000	SLOC	COBOL
SLOC Based Project D	10000	SLOC	Pascal
6 Core Data Structures	240	UFP	3rd Ge…

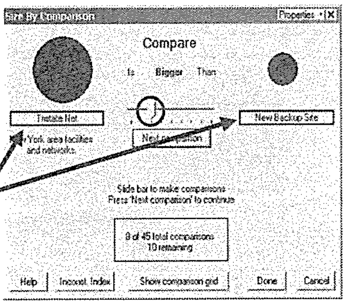
Estimated Items

Name	Size	Metric	Language	Description
New Backup Site	272	UFP	C++	
Centric Operations Site	315	UFP	C++	

Current Selection
Language: C++
Legacy: New Development
Scoped by comparison
Least Likely Most
Unadjusted Function Points
315 315 315

SEER-AccuScope Items Sized: 11 out of 11 Sizing Origin: Estimated

Step 3a: Compare Known & Unknown Items



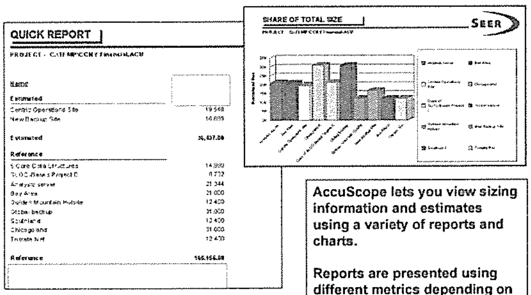
Size By Comparison

Slide bar to make comparisons
Press 'Next comparison' to continue

9 of 45 total comparisons
10 remaining

Expert judgment establishes the relative size between 'Tristate Net' and 'New Backup Site'.

Step 4: View / Use Results



QUICK REPORT

PROJECT - CATEMPCCRY FinancialACE

Name

Estimated

Centric Operations Site	19,548
New Backup Site	16,889
Estimated	36,437.00

Reference

5 Core Data Structures	14,550
SLOC Based Project C	8,732
Analysis server	21,344
Bay Area	31,000
Golden Mountain Hubsite	12,400
Global backup	31,000
Southland	12,400
Chicagoland	31,000
Tristate Net	12,400
Reference	165,156.00

AccuScope lets you view sizing information and estimates using a variety of reports and charts.

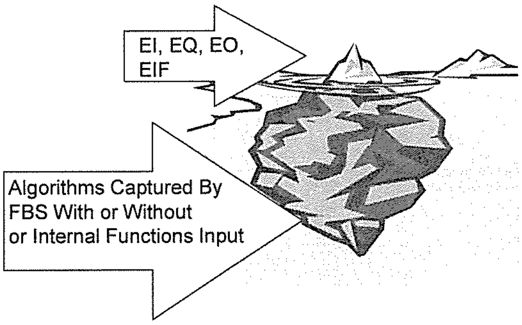
Reports are presented using different metrics depending on the size metric (source lines of code, function points, etc.) you are working with.

Line 대 Function Point 변환기준 개선

- Galorath Does Not Recommend Arbitrary Line To Function Point Conversions
- Galorath Early Work Showed Published Ratios Were Significant Problems
- Language

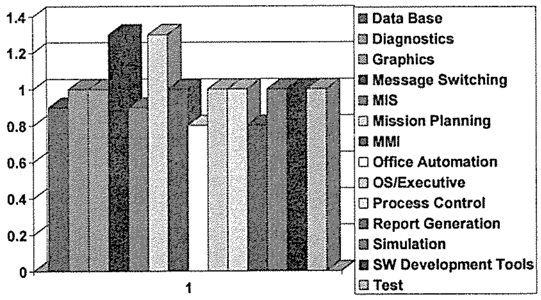
Language	Jones	Galorath Likely
C	128	61
FORTRAN	105	58
COBOL	105	61
PASCAL	91	71
PL/I	80	71
ADA	71	73
Menu Driven Generators	16	15

기존 기능점수에 반영되지 않은 요소를 추가



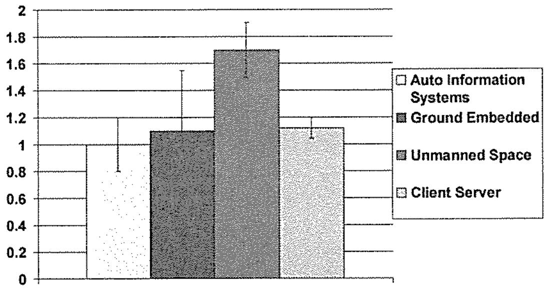
EI, EQ, EO, EIF
Algorithms Captured By FBS With or Without or Internal Functions Input

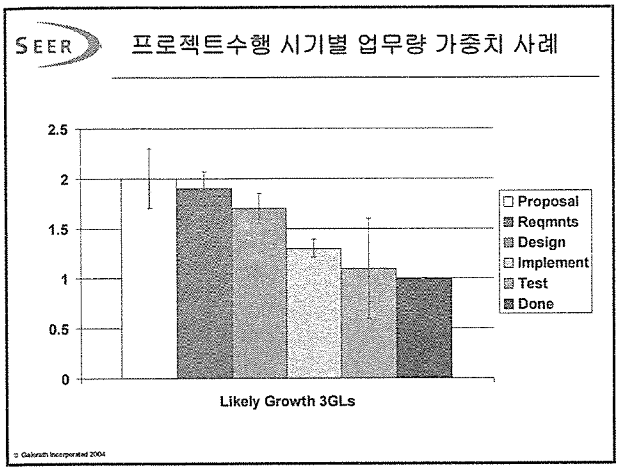
응용영역별 기능점수당 업무량 가중치 사례



Note: These Are Examples. SEER-SEM Contains Complete List

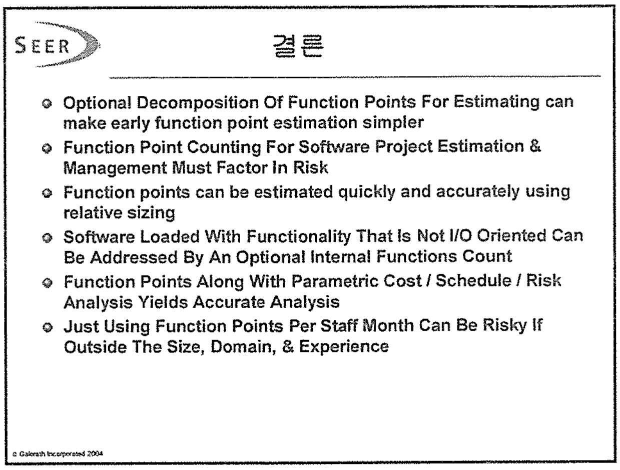
플랫폼별 기능점수당 업무량 가중치 사례





프로젝트수행 시기별 업무량 가중치 사례

© Galorath Incorporated 2004



결론

- Optional Decomposition Of Function Points For Estimating can make early function point estimation simpler
- Function Point Counting For Software Project Estimation & Management Must Factor In Risk
- Function points can be estimated quickly and accurately using relative sizing
- Software Loaded With Functionality That Is Not I/O Oriented Can Be Addressed By An Optional Internal Functions Count
- Function Points Along With Parametric Cost / Schedule / Risk Analysis Yields Accurate Analysis
- Just Using Function Points Per Staff Month Can Be Risky If Outside The Size, Domain, & Experience

© Galorath Incorporated 2004