Results of eServices Product Sizing Metrics Correlations

Yue Chen, Barry Boehm, Ray Madachy, Winsor Brown
Center for Software Engineering
University of Southern California
941 W. 37th Place, SAL Room 328
Los Angeles, CA 90089-0781
{yuec, boehm, madachy}@sunset.usc.edu
October 21, 2003

Presentation Outline

- Motivation of UML sizing
- Selection of projects
- Counting methodology
- Experiment results and analysis
- Conclusion
- Open Issues

Why UML Sizing?

- Solid information source about software size and complexity
- Countable design element
- UML is the most popular system modeling language
- Programming language independent

UML Diagrams

MAJOR	VIEW	DIAGRAMS
Structural	Static view	Class diagram
	Use Case view	Use case diagram
	Implementation view	Component diagram
	Deployment view	Deployment diagram
Dynamic	Interaction view	Sequence diagram
		Collaboration diagram
	Activity view	Activity diagram
	State machine view	State transition diagram
Management	Model management view	Class diagram, package, subsystem, etc.

How projects are selected?

All selected projects are

- from 2001 – 2003 USC CSCI577(b) class projects
- targeted in building eService Applications
- development intensive projects (none COTS intensive)

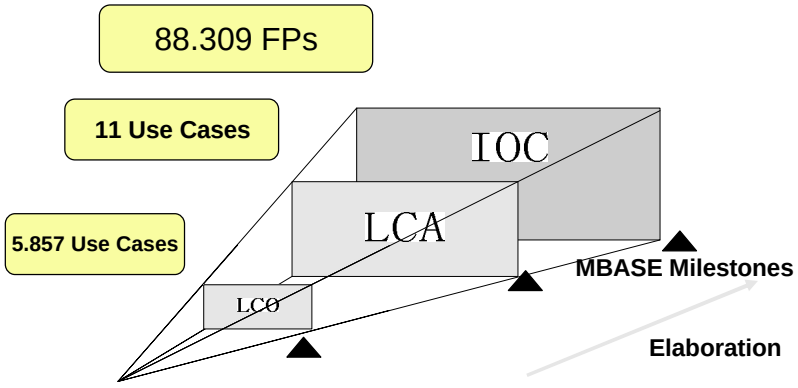
List of 14 Projects

CODE	TITLE	LANGUAGE	DOMAIN
Sp03t03	UML2Web	Python	Web
Sp03t06	Pilot Web Based Geotechnical Virtual Data Center	JSP	Web
Sp03t09	Conference Trip Planning System	HTML, PHP, SQL	Web
Sp03t13	Quality Information Management System for 577 Course	JSP	Web
Sp03t14	Caroline's Closet	ASP	Web
Sp02t01	Dental Library New Booklist	Perl	Web
Sp02t06	ISD Interactive Web Based Contract Management System	PhP, HTML	Web
Sp02t15	Strategic Risk-Value Assessment Tool	Java, JSP	Web
Sp02t19	Opportunity Tree Framework	JavaScript	Web
Sp01t01	Station Data Project for the Web	HTML/JAVA	Web
Sp01t03	Pathology Image Search Engine	JSP/JAVA	Web
Sp01t08	Full-text Titles Database	HTML/JAVA	Web
Sp01t14	Access & Display Archive Image Composer	JSP/ JAVA/ HTML/ SQL/ CDML	Web
Sp01t17	Web Mail	JAVA/ HTML/ JavaScript/ JSP	Web

Counting Methodology

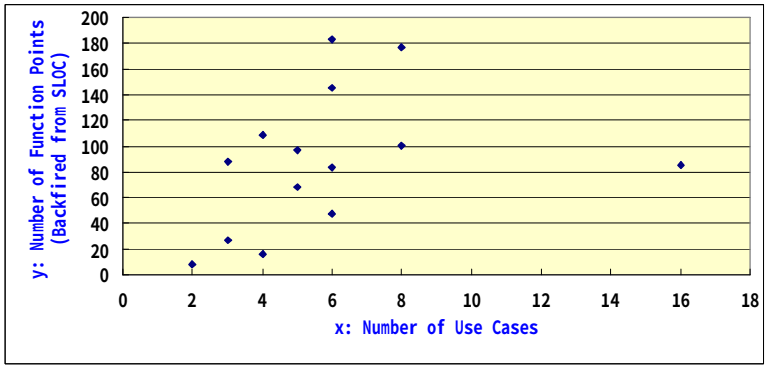
- The code counter tool used for counting is the Code Counter Pro by *Geronesoft*. The result is justified into Logical SLOC by comparing LSLOC/PSLOC ratio.
  Link: http://www.investph.com/geronesoft/
- The number of function point is backfired from SLOC based on the "Function Point Language Gearing Factors " provided by QSM.
  Link: http://www.qsm.com/FPGearing.html
- The project UML statistics are counted from the USC CSCI 577 course archives (2001 – 2003, 14 projects)
  Link: http://www.cebase.org:444/usc/mbase_projects_archive/archive.html

Level of Design Detail Matters



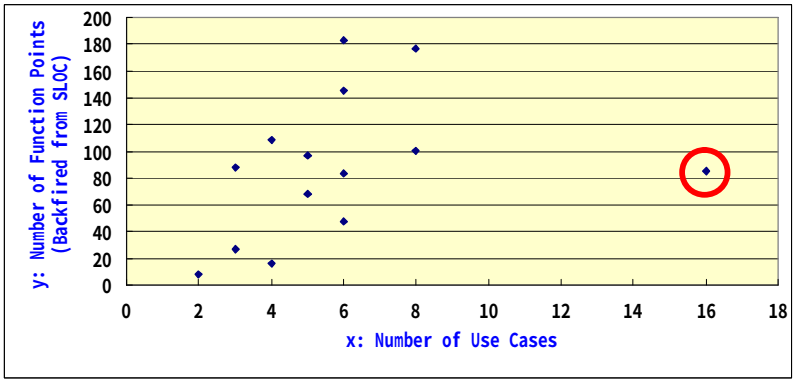
[Note] The numbers in the above diagram are calculated from the counting average from 14 USC CSCI 577 software engineering course projects.

Use Cases vs. Function Points



- **The number of *Function Point* is backfired from the counting result of *SLOC***
- **The number of use cases is counted from the *OCD* document in the project "*Final Deliverables*" package**
- **The code counter tool used for counting is the *Code Counter Pro* by *Geronesoft.***

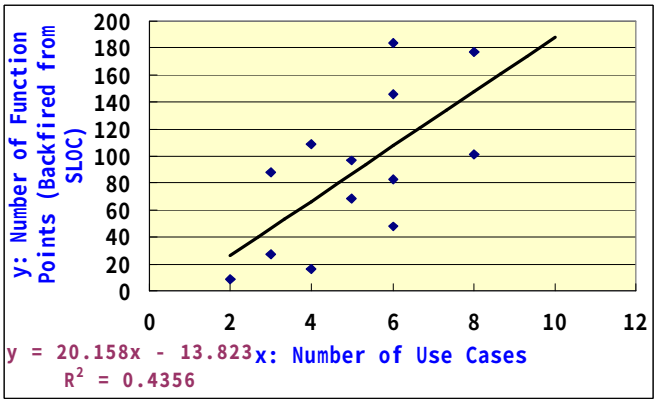
Use Cases vs. Function Points



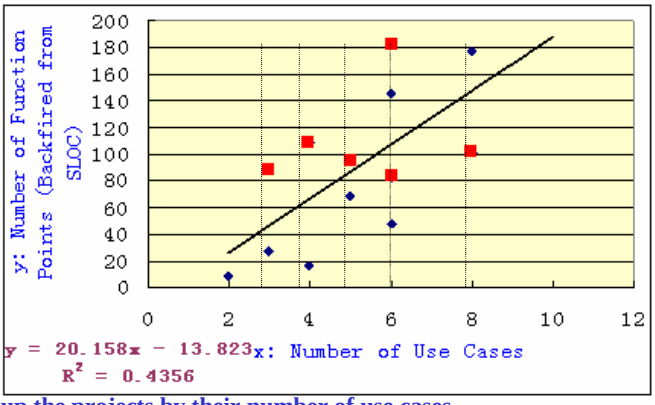
Why there is an outlier?

The use case diagram of this project is over detailed. There is an inconsistence between the diagram and the documentation: there are 16 use cases but only 9 of them are documented with implementation detail.

Trend Line without the Outlier

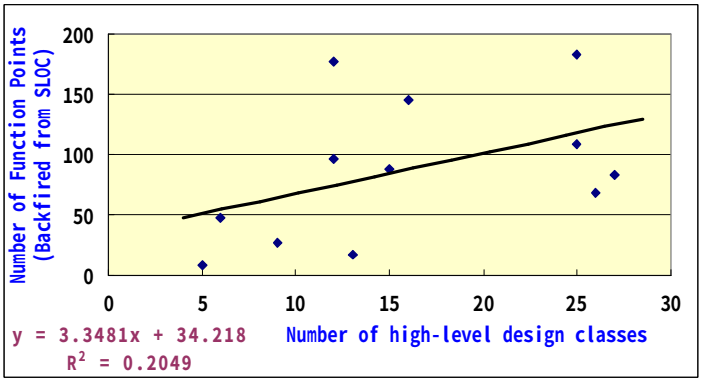


Effect of Project Complexity



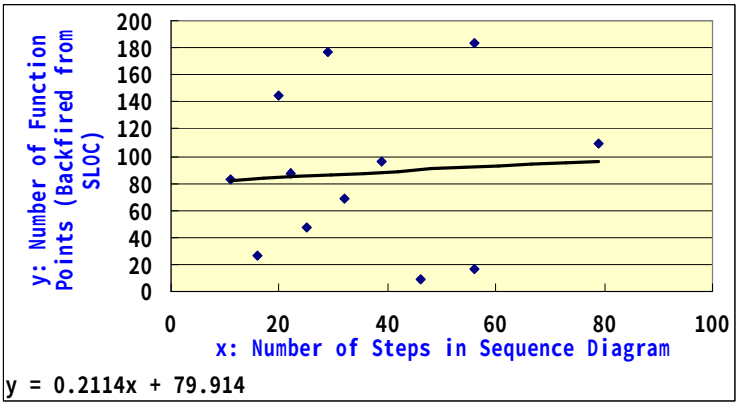
- Group the projects by their number of use cases.
- ■ : the projects with greater value of the number of sequence-diagram-steps per use case comparing with its other group members.

Comparison: FP vs. High-level Design Classes



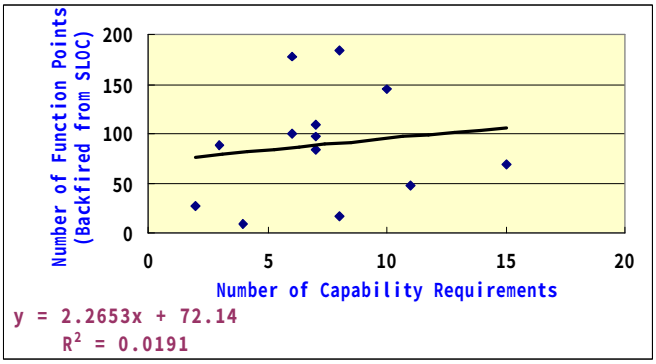
[Note] The number of **high-level design classes** is counted from the SSAD document in the project "*Final Deliverables*" package.

Comparison: FP vs. Number of Steps in Sequence Diagram



[Note] The number of capability requirement is counted from the SSAD document in the project "*Final Deliverables*" package.

Comparison: FP vs. Number of Capability Requirements



[Note] The number of capability requirement is counted from the SSRD document in the project "*Final Deliverables*" package.

R^2 Comparison

R^2	y=a*x+b (Linear)	y = log(x) (Logarithmic)	$y = a^x$ (Exponential)	$y = x^a$ (Polynomial)
FP v.s. # of Use Case	0.436	0.442	0.476	**0.548**
FP v.s. # of classes	0.205	0.277	0.313	0.440
FP v.s. # of capability requirements	0.019	0.071	0.057	0.103

Conclusions

- Data shows the strongest correlation between the number of use cases and the number of backfired FPs.
- The polynomial model fits best

  (Number of FP) ~ 4.7363(Number of Use Cases)$^{1.6938}$, R^2 = 0.548

- The number of sequence diagram steps per use case seems to be a project complexity indicator

Open Issues

- The team's documentation style
  - A complex project documentation cannot conclude a complex project implementation.
- The accuracy of backfiring SLOC to Function Point
- More project data points

Thanks!