THE (MIS)USE OF SUBJECTIVE PROCESS MEASURES IN SOFTWARE ENGINEERING

Jon D. Valett

SOFTWARE ENGINEERING BRANCH
Code 552
NASA/Goddard Space Flight Center
Greenbelt, Maryland 20771

Steven E. Condon

COMPUTER SCIENCES CORPORATION
GreenTec II—10110 Aerospace Road
Lanham-Seabrook, Maryland 20706

WHAT ARE SUBJECTIVE PROCESS MEASURES?

A variety of measures are used in software engineering research to develop an understanding of the software process and product. These measures fall into three broad categories: quantitative, characteristics, and subjective. Quantitative measures are those to which a numerical value can be assigned, for example effort or lines of code (LOC). Characteristics describe the software process or product; they might include programming language or the type of application. While such factors do not provide a quantitative measurement of a process or product, they do help characterize them. Subjective measures (as defined in this study) are those that are based on the opinion or opinions of individuals; they are somewhat unique and difficult to quantify.

Capturing of subjective measure data typically involves development of some type of scale. For example, "team experience" is one of the subjective measures that were collected and studied by the Software Engineering Laboratory (SEL). Certainly, team experience could have an impact on the software process or product; actually measuring a team's experience, however, is not a strictly mathematical exercise. Simply adding up each team member's years of experience appears inadequate. In fact, most researchers would agree that "years" do not directly translate into "experience." Team experience must be defined subjectively and then a scale must be developed—e.g., high experience versus low experience; or high, medium, low experience; or a different or more granular scale. Using this type of scale, a particular team's overall experience can be compared with that of other teams in the development environment.

Defining, collecting, and scaling subjective measures is difficult. First, precise definitions of the measures must be established. Next, choices must be made about whose opinions will be solicited to constitute the data. Finally, care must be given to defining the right scale and level of granularity for measurement.

WHY DO SOFTWARE ENGINEERS NEED SUBJECTIVE MEASURES?

Despite the difficulties inherent in working with subjective measures, many researchers propose that the software process and product can not be characterized fully without them. Early work by Walston and Felix[1] used subjective data for characterizing software. Intermediate COCOMO[2] uses 16 subjective cost drivers for estimating software cost. These subjective measures range from "amount of experience with the development programming language" to "product complexity." For a given project, each of the 16 factors is rated and used to develop the basic cost estimate. The expectation is that inclusion of these factors will yield a more precise/accurate cost estimate. In fact, almost all cost models use some subjective factors.

In addition to cost modeling, software engineering researchers use subjective measures to help quantify other aspects of the software process. For example, they might try to determine if the team experience factor has any impact on productivity or reliability. In developing a reliability model, they might look at the quality of the team's code reading. Subjective measures can also be used in defining software domains. In this application, a subjective measure might be considered a defining factor in placing particular software in one domain versus another. Projects that use formal structured analysis, for example, may be in a different domain from those that use other methods.

This research examines the use of subjective measures in software engineering experimentation. In the sections that follow, this paper discusses the early experiences of the SEL collecting and applying subjective measure data, looks at refinements the SEL made to their collection and analysis process, and then reports on more recent SEL studies using subjective data. Some general recommendations are made for the collection and use of subjective data based on lessons learned in the SEL.

THE SEL AND SUBJECTIVE MEASURES

The SEL is a research organization that supports the Flight Dynamics Division of NASA/Goddard Space Flight Center. Its purpose is to investigate the effectiveness of software engineering technologies applied to the development of flight dynamics software.

The SEL collects a variety of data from application software projects for use in its research and experiments. These data include information on effort, size, computer resources, project characteristics, and a number of subjective measures. (For a complete description of the data collected see Reference 3.) The SEL began collecting subjective measures data in 1977. The primary goals for these data were to validate the models of other software engineering researchers and to fully characterize the SEL environment. As with many early SEL data collection efforts, an attempt was made in this case to collect every possible piece of data. On each project, over 300 individual subjective measures were collected.

For each measure, managers gave an opinion expressed as a rating based on a 0-5 scale. The data were not validated/cross-checked in any way before being stored in the SEL database. No one else examined the ratings given or tried to provide consistency across projects. Furthermore, the 0-5 ratings were not defined. Thus, for the same measure on the same project, two different individuals might have given different ratings. While this was somewhat minimized because there were very few people providing the data, the data were still inconsistent. Also, due to the lack of precise definitions for ratings, inconsistency was possible not only amongst data providers but also from project to project and from year to year. That is, because of changing perceptions, similar projects may have been given different ratings. Nevertheless, these data were used by the SEL in a variety of experiments, two of which are detailed below.

Early Uses of Subjective Measures

One early experiment using subjective measures was the development of a meta-model for software development resource expenditures.[4] The goal of the experiment was to develop a cost model that included subjective process measures. First, the subjective data from the SEL database were converted from the 0-5 scale to a binary (high/low) scale for use in the experiment. Second, the researchers selected one manager who was familiar with all the projects as a source for establishing consistency across the projects.

Using data from 17 projects, the researchers developed a baseline cost model that related effort to LOC. They examined the impact on cost of 71 different subjective measures to determine if any of them showed a significant relationship to the cost of the project. No significant correlation was found. The data proved to be too detailed to really determine if there was any impact. While the researchers were able to find some correlation between certain measures and cost, it was not consistent. The researchers then applied a grouping technique to the measures, converting the 71 measures into three groups. This allowed them to build new, broader-based subjective measures. Using these three measures they built a new cost model which they later confirmed against new projects that were similar to those in the data set.

Two main points emerge from reviewing this experiment:

- *Be wary of "looking for correlations."* While these researchers found some correlations when using the detailed data, they proved to be inconsistent. In almost any experiment using subjective data some correlations may exist, but they must be repeatable to be significant.
- *Collecting lots of data does not guarantee lots of results.* In this experiment the vast amount of data collected had to be converted to a much less detailed set.

In a second experiment using subjective measures, SEL researchers sought to determine the effect of modern programming practices (MPPs) on productivity and reliability.[5] Again, the subjective measures data in the SEL database were used after being converted to a binary scale and combined into groups. However, the grouping method used in this experiment differed from the method used in the previous experiment. Various subjective measures were combined with quantitative data to predefine MPPs such as structured coding and tool use. Then, analyzing data from 22 projects, the researchers tested the effects of MPPs on productivity and reliability. No correlation was shown on productivity, while quality of documentation, amount of quality assurance, and quality of code reading did have an impact on error rate. Unfortunately, these results were never confirmed over other data sets.

Major lessons on subjective measures from this study are:

- *Detailed subjective data probably are not useful.* Having over 300 different subjective measures actually proved to be less useful than having fewer, more general categories of subjective information.
- *To validate results using subjective data, confirm the results across multiple data sets.*

Refining SEL Subjective Data Collection

In 1987, the SEL (recognizing the difficulty with collecting and using over 300 detailed subjective measures) set out to significantly reduce the data set. Based on the experience of other researchers and the specific experience of the SEL, a new set of 36 measures was defined. These data continue to be collected today.

Subjective measure data are now provided by project leads. At the end of each project, the project lead completes a questionnaire that uses a 1-5 scale. (The questionnaire is included as an appendix.) The opinions of the project lead are presumed to be accurate; no other validation or cross-checking of the data is done. This data collection policy still allows bias and potential inconsistency within the data as people with different perspectives and experiences might give the same project different scores. Two experiments using the newer subjective data are discussed below.

Recent Experiences with Subjective Measures

Recently, a study was conducted in which the 36 subjective measures were applied to a basic cost model. This was done as part of a larger effort to build a specific cost model for the SEL environment.[6] In this study the researchers used the measures as they were recorded in the SEL database. They developed a basic cost model and then attempted to improve that model by adding various subjective measures. On the initial data sets used, some of the measures did appear to improve the cost models, but when the researchers tried to validate the models using different data sets (from similar projects) they were unable to duplicate the results. In fact, they found similar improvements in the models when they substituted random data for the actual subjective measures data. Given these results, the researchers concluded that the current subjective data should not be used as a factor in projecting cost.

Two lessons learned from this experience:

- *Collecting data on a 1-5 scale is probably not optimal.* Distinguishing each rating, for example a "2" versus a "3," is difficult. In the past, when these data have been used in analysis they have been converted to a binary scale. The scale should be reduced either when the data are collected or when they are used.
- *Results should be confirmed over multiple data sets.* This has been pointed out before, but it bears repeating. In too many instances researchers have come to conclusions based on one set of projects without checking out the results on other similar projects.

Another study was conducted (specifically for this report) with the goal of determining the impact of subjective measures on effort, errors, and changes. Data were converted to a binary scale. Also, the analytic method used assumed that the 36 measures were not independent. (The previous study did not address the dependency of the data.) For any set of projects, a linear model was built relating the size of a project to a particular measure, such as changes. Then a set of subjective measures that may have had an impact on the chosen measure was identified. From that set, the factors that were most likely to have had an impact and those that best represented the dependent set of measures were added to an enhanced linear model. Attempts to validate these models against multiple similar data sets showed little or no consistency.

Based on this study and the others discussed, it appears that even the conservative use (i.e., using a binary scale and incorporating data dependency) of the subjective measures data collected by the SEL is of questionable value. While previous analyses of the data showed some promise, recent experiences have been less successful.

MISUSES AND USES OF SUBJECTIVE MEASURES

Based on these findings, SEL researchers have questioned the value of collecting these data. Although the data may not be viable for rigid statistical analyses, they can be important tools for environment characterization and research planning purposes. When working with subjective measures, the following guidelines should be considered:

- *Be cognizant of the data collection mechanism and the extent to which the data are validated.* Make no assumptions concerning the accuracy and validity of the data.
- *When defining subjective measures for collection, less is usually better.* Collecting a wide variety of data without a plan for their use is pointless.
- *Use subjective measures to spot trends and set goals for more detailed experiments.* General subjective measures can be a good place to start when setting goals for research. This is probably the best way to use loosely defined, nonvalidated subjective measures such as those collected by the SEL.

Given the somewhat limited usefulness of the SEL's subjective measure data, the SEL might be expected to abandon collection of subjective data. Subjective information, however, is important for understanding an environment and it provides a context for data analysis. When designing experiments or studies, a researcher needs to examine subjective information about a project to decide if that project is appropriate for inclusion in a particular study. That information might, however, be more likely found in project documents (e.g., lessons learned reports) than in ranked questionnaire responses.

Rather than abandon subjective measure data collection, the SEL needs to define a set of subjective measures that accurately captures the critical elements of the local environment. From there, a set of goals for the subjective measures must be identified and a set of questions generated that precisely defines the measures for the local environment. The last step would be to develop a methodology for collecting and validating the data. If such steps are taken, the validity of the subjective measures data could be improved and their usefulness in the SEL's ongoing process improvement program could be reexamined.

REFERENCES

1. Walston, C., and C. Felix, "A Method of Programming Measurement and Estimation," *IBM Systems Journal* 16, Number 1, 1977

2. Boehm, B. W., *Software Engineering Economics*, Prentice-Hall, Englewood Cliffs, New Jersey, 1981

3. Heller, G., J. Valett, and M. Wild, *Data Collection Procedures for the Software Engineering Laboratory (SEL) Database*, SEL-92-002, Software Engineering Laboratory, Greenbelt, Maryland, 1992

4. Basili, V. R., and J. W. Bailey, "A Meta-Model for Software Development Resource Expenditures," *Proceedings of the Fifth International Conference on Software Engineering*, IEEE Computer Society Press, New York, New York, 1981

5. Card, D. N., F. E. McGarry, and G. T. Page, "Evaluating Software Engineering Technologies," *IEEE Transactions on Software Engineering*, July 1987

6. Condon, S., et al., *Cost and Schedule Estimation Study Report*, SEL-93-002, Software Engineering Laboratory, Greenbelt, Maryland, 1993

APPENDIX—SEL SUBJECTIVE DATA COLLECTION QUESTIONNAIRE

SUBJECTIVE EVALUATION FORM

Name: ______________________

Project: ______________________ Date: ______________________

Indicate response by circling the corresponding numeric ranking.

I. PROBLEM CHARACTERISTICS

1. Assess the intrinsic difficulty or complexity of the problem that was addressed by the software development.

1	2	3	4	5
Easy		Average		Difficult

2. How tight were schedule constraints on project?

1	2	3	4	5
Loose		Average		Tight

3. How stable were requirements over development period?

1	2	3	4	5
Loose		Average		High

4. Assess the overall quality of the requirements specification documents, including their clarity, accuracy, consistency, and completeness.

1	2	3	4	5
Low		Average		High

5. How extensive were documentation requirements?

1	2	3	4	5
Low		Average		High

6. How rigorous were formal review requirements?

1	2	3	4	5
Low		Average		High

II. PERSONNEL CHARACTERISTICS: TECHNICAL STAFF

7. Assess overall quality and ability of development team.

1	2	3	4	5
Low		Average		High

8. How would you characterize the development team's experience and familiarity with the application area of the project?

1	2	3	4	5
Low		Average		High

9. Assess the development team's experience and familiarity with the development environment (hardware and support software).

1	2	3	4	5
Low		Average		High

10. How stable was the composition of the development team over the duration of the project?

1	2	3	4	5
Loose		Average		High

FOR LIBRARIAN'S USE ONLY

Number: ______________________ Entered by: ______________________

Date: ______________________ Checked by: ______________________

8201G(13)-29

NOVEMBER 1991

APPENDIX—SEL SUBJECTIVE DATA COLLECTION QUESTIONNAIRE

SUBJECTIVE EVALUATION FORM

III. PERSONNEL CHARACTERISTICS: TECHNICAL MANAGEMENT

11. Assess the overall performance of project management.
    1 Low 2 3 Average 4 5 High

12. Assess project management's experience and familiarity with the application.
    1 Low 2 3 Average 4 5 High

13. How stable was project management during the project?
    1 Low 2 3 Average 4 5 High

14. What degree of disciplined project planning was used?
    1 Low 2 3 Average 4 5 High

15. To what degree were project plans followed?
    1 Low 2 3 Average 4 5 High

IV. PROCESS CHARACTERISTICS

16. To what extent did the development team use modern programming practices (PDL, top-down development, structured programming, and code reading)?
    1 Low 2 3 Average 4 5 High

17. To what extent did the development team use well-defined or disciplined procedures to record specification modifications, requirements questions and answers, and interface agreements?
    1 Low 2 3 Average 4 5 High

18. To what extent did the development team use a well-defined or disciplined requirements analysis methodology?
    1 Low 2 3 Average 4 5 High

19. To what extent did the development team use a well-defined or disciplined design methodology?
    1 Low 2 3 Average 4 5 High

20. To what extent did the development team use a well-defined or disciplined testing methodology?
    1 Low 2 3 Average 4 5 High

IV. PROCESS CHARACTERISTICS

21. What software tools were used by the development team? Check all that apply from the list that follows and identify any other tools that were used but are not listed.
    - ☐ Compiler
    - ☐ Linker
    - ☐ Editor
    - ☐ Graphic display builder
    - ☐ Requirements language processor
    - ☐ Structured analysis support tool
    - ☐ PDL processor
    - ☐ ISPF
    - ☐ SAP
    - ☐ CAT
    - ☐ PANVALET
    - ☐ Test coverage tool
    - ☐ Interface checker (RXVP80, etc.)
    - ☐ Language-sensitive editor
    - ☐ Symbolic debugger
    - ☐ Configuration Management Tool (CMS, etc.)
    - ☐ Others (identify by name and function)

22. To what extent did the development team prepare and follow test plans?
    1 Low 2 3 Average 4 5 High

6201G(13)-30

SUBJECTIVE EVALUATION FORM

IV. PROCESS CHARACTERISTICS (CONT'D)

23. To what extent did the development team use well-defined and disciplined quality assurance procedures (reviews, inspections, and walkthroughs)?

1	2	3	4	5
Low		Average		High

24. To what extent did development team use well-defined or disciplined configuration management procedures?

1	2	3	4	5
Low		Average		High

V. ENVIRONMENT CHARACTERISTICS

25. How would you characterize the development team's degree of access to the development system?

1	2	3	4	5
Low		Average		High

26. What was the ratio of programmers to terminals?

1	2	3	4	5
8:1	4:1	2:1	1:1	1:2

27. To what degree was the development team constrained by the size of main memory or direct-access storage available on the development system?

1	2	3	4	5
Low		Average		High

28. Assess the system response time: were the turnaround times experienced by the team satisfactory in light of the size and nature of the jobs?

1	2	3	4	5
Poor		Average		Very Good

29. How stable was the hardware and system support software (including language processors) during the project?

1	2	3	4	5
Low		Average		High

30. Assess the effectiveness of the software tools.

1	2	3	4	5
Low		Average		High

VI. PRODUCT CHARACTERISTICS

31. To what degree does the delivered software provide the capabilities specified in the requirements?

1	2	3	4	5
Low		Average		High

32. Assess the quality of the delivered software product.

1	2	3	4	5
Low		Average		High

33. Assess the quality of the design that is present in the software product.

1	2	3	4	5
Low		Average		High

34. Assess the quality and completeness of the delivered system documentation.

1	2	3	4	5
Low		Average		High

35. To what degree were software products delivered on time?

1	2	3	4	5
Low		Average		High

36. Assess smoothness or relative ease of acceptance testing.

1	2	3	4	5
Low		Average		High

6201G(13)-31

The (Mis)use of Subjective Process Measures in Software Engineering

Jon. D. Valett
NASA/GSFC

Steve Condon
CSC

G218.001

Categories of Measurement Data

Quantitative	Characteristics	Subjective
- Effort	- Programming Language	- Team experience
- LOC	- Platform	- Requirements stability
- Computer use	- Application	- Degree of MPP
• • •	• • •	• • •

Subjective Measures --
those that are based on the opinion of individuals

How Should Subjective Measures be Used in Software Engineering?

G218.002

Need for Subjective Measures

- Help to Quantify the Software Process
  - Does team experience impact productivity?
  - Do Modern Programming Practices (MPPs) impact the development process and product?
- Improve Models of Software Process and Product
  - Error Rate = X * Developed LOC - Y * Quality of Code Reading
  - Intermediate COCOMO
- Define Software Domains
  - Are projects that use structured analysis different from those that don't?

G218.003

Subjective Measures

- There are many subjective measures
  e.g.
  - Team experience
  - Management stability
  - Machine availability
  - Quality of tool set
  - Schedule constraint
  - Product complexity

  etc.
- There have been many proposed uses -
  - Walston and Felix
  - COCOMO
  - Other Cost Models
  - Domain Analysis

G218.004

The SEL and Subjective Measures

Beginning in 1977 the Software Engineering Laboratory (SEL) began collecting subjective measures

Philosphy -

Validate models of other researchers

Fully characterize the environment

Sumary

What Data?	Collect Everything (over 300 individual measures)
Who Provides?	Managers rate
How Collected?	After project completion Use 0-5 scale
How Clarified/Validated	None

G218.005

Use of Subjective Measures

"The Meta-Model for Software Development Resource Expenditures*"

Goal:
Develop a cost model that incorporates subjective process measures

Subjective Measures:
- Converted to binary scale
- Validated measures using 1 manager as source
- Converted detailed data into three groups

Process:

Develop baseline model 1.17 Effort = .72 DevLOC + 3.4 using 17 projects ➡ Attempt to incorporate 71 subjective measures **Yielded meaningless results** ➡ Converted 71 measures to 3 groups ➡ Created new model effort = initial model + effort multipliers

Result:
Model confirmed using new projects similar to those in data set

Lessons:
- If you look hard enough you may find some correlations
- A lot of data does not generate a lot of results

* Bailey and Basili 1981

G218.006

Use of Subjective Measures

"Evaluating Software Engineering Technologies*"

Goal:
Do MPPs affect productivity and reliability?

Subjective Measures:
- Converted to binary scale
- Combined data into groups

Process:

Combined subjective measures with quantitative data on 22 projects to define MPPs ➡ Tested affects of MPPs such as
Quality assurance
Tool use
Structured Code
Code reading
•
•
• ➡ No findings on cost

Error Rate affected by -
Documentation
Quality assurance
Code reading

Result:
Not confirmed over other data sets (within the same domain) - conclusions questionable

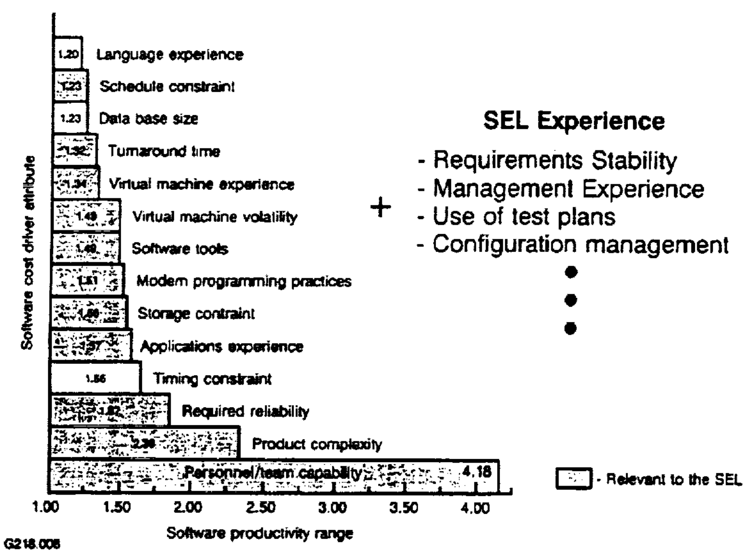
Lessons:
- Confirm results over multiple similar data sets
- A lot of data does not generate a lot of results

* Carc, McGarry, Page 1987

G218.007

Reducing the Measure Set

Boehm's Software Engineering Economics ·



G218.008

SEL Experience

- Requirements Stability
- Management Experience
- Use of test plans
- Configuration management

•
•
•

\+ ... = 36 measures

Current Subjective Measures (1987 - Present)

Philosphy -

Determine the impact of key measures on the software process and product

Characterize the environment

Sumary

What Data?	36 General subjective measures
Who Provides?	Project leads
How Collected?	After project completion Use 1-5 scale Survey form
How Clarified/Validated	None

The SEL continues to collect high level subjective measures

G218.009

Use of Subjective Measures

"Cost Estimation Study"

Goal:
Improve a basic cost model using subjective measures

Subjective Measures:
- Used the 1-5 ratings
- Used multiple data sets

Process:

Start with basic cost model
Effort = DevLOC/3.2

➡ Use subjective data to find new models

$$\text{effort} = \frac{\text{DevLOC}}{3.2^{*}[1 - (\text{weight}^{*}\text{subj.measure})]}$$

➡ Some measures improve model

➡ Validate
- using multiple data sets
- try random numbers

Results:
- Enhanced models inconsistent over multiple data sets
- Random numbers improved models as well as the real data

Lessons:
- Tend toward conservative use of measures (1-5 scale too detailed)
- Carefully evaluate all results

* Condon, Regardie 1993

G218 010

Use of Subjective Measures

"Impact of Subjective Measures on Effort, Errors, and Changes"

Goal:
Are there subjective measures that impact effort, errors, and changes?

Subjective Measures:
- Converted to binary scale
- Assumed dependency in the data
- Used multiple data sets

Process:

Develop a linear model
Changes = X * DevLOC

→ Find subjective measures that may impact
e.g.
- Quality of design
- Quality of documentation
•
•
•

→ Develop enhanced linear model
changes = X * DevLOC - Y * Subjective measure

→ Validate
- Using multiple data sets

Result:
Little or no consistency found among data sets

Lessons:
- Even conservative use of data is questionable
- The 36 measures are not independent

G218.011

Misuses of Subjective Measures

- Don't search for correlations, because you will find at least one
- Don't collect too much data without understanding how to use it
- Don't go beyond the validity and consistency of your data
- Don't rely on on-line data - except to spot trends/set goals

The measures contain no miracle answers. They are only one tool.

G218.012

Subjective Measures ≠ Subjective Information

- Subjective Information Provides Context for Analysis
  - Lessons learned documents
  - Project annotations
- Set Goals for Subjective Information
- Subjective Information Transformed into Subjective Measures by
  - Local definitions
  - Using consistent data collection methods

Subjective information is critical to understanding an environment, but don't think it is easy

G218.013