APPLICATION OF A QUANTITATIVE SOFTWARE METRICS ASSESSMENT PROCESS TO MILITARY SOFTWARE DEVELOPMENT PROGRAMS

Presented by:

John J. McGarry
Cheryl L. Jones
Naval Undersea Warfare Center
Newport, RI

ELECTRO INTERNATIONAL 1994
Hynes Convention Center
Boston, MA
May 10 - 12, 1994

APPLICATION OF A QUANTITATIVE SOFTWARE METRICS ASSESSMENT PROCESS TO MILITARY SOFTWARE DEVELOPMENT PROGRAMS

by
John J. McGarry
Cheryl L. Jones
Naval Undersea Warfare Center

ABSTRACT

The objective assessment of DOD Weapons System and Automated Information System software development programs is becoming increasingly dependent on the application and analysis of software process and product related metrics. Experience in the actual application of software metrics to military programs has shown that a key requirement for a practical software measurement program is a consistently implemented and repeatable quantitative software development assessment process applied across the software development life cycle. Within the defined software metrics process software attributes are continuously measured and evaluated with respect to development program characteristics, constraints, and objectives, and the results are subsequently integrated into an overall assessment of the software development program. A key part of the defined software metrics assessment process is the relationship of software development process attributes and measurement results to expected and observed product characteristics. Implementation of the metrics process has shown that effective development processes yield high quality software products.

DESCRIPTION

Figure 1 provides an overview of the software metrics assessment process which has been successfully applied to a number of large scale military software development programs. Each of the four major assessment sub-processes, Software Issue Definition, Software Attribute Measurement, Software Indicator Generation, and Software Quantitative Assessment, are supported by a number of tools and analysis methodologies which combine to provide a flexible but disciplined process for identifying, evaluating, and correcting software development issues and problems. Each of the four sub-processes is iterated with respect to program characteristics and development issues until an objective integrated assessment can be identified and validated.

QUANTITATIVE SOFTWARE ASSESSMENT PROCESS

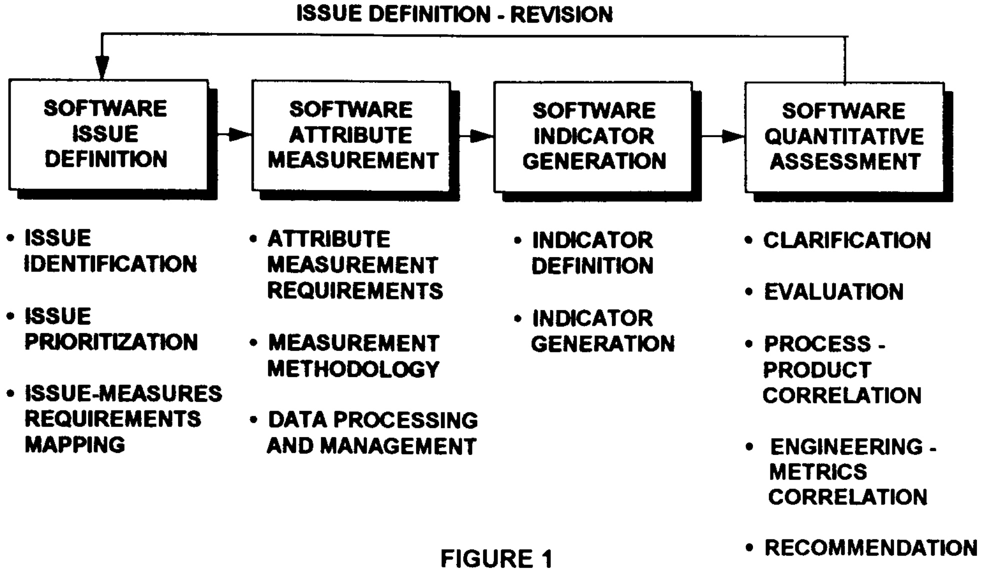
ISSUE DEFINITION - REVISION
SOFTWARE ISSUE DEFINITION
SOFTWARE ATTRIBUTE MEASUREMENT
SOFTWARE INDICATOR GENERATION
SOFTWARE QUANTITATIVE ASSESSMENT
• ISSUE IDENTIFICATION
• ISSUE PRIORITIZATION
• ISSUE-MEASURES REQUIREMENTS MAPPING
• ATTRIBUTE MEASUREMENT REQUIREMENTS
• MEASUREMENT METHODOLOGY
• DATA PROCESSING AND MANAGEMENT
• INDICATOR DEFINITION
• INDICATOR GENERATION
• CLARIFICATION
• EVALUATION
• PROCESS - PRODUCT CORRELATION
• ENGINEERING - METRICS CORRELATION
• RECOMMENDATION

FIGURE 1

KEY CONCEPTS

A number of key concepts are applicable in the definition and implementation of the software metrics assessment process. These include the following:

- **The metrics process is based upon defined quantitative measures of software process and product attributes measured consistently across the software development program life cycle. The actual measures applied for different programs within a functional domain vary with different program characteristics. Measurement methodologies should be consistent, however, for a given metric applied within the program. Methods of counting planned and actual Source Lines of Code, for example, should be defined early in the program and remain unchanged.**

- **The metrics process is flexible and tailorable to distinct software development program characteristics, objectives, and limitations. It can and should be adapted to address different software development process models and implementations.**

- The overall metrics assessment process is issue driven. The specific program development issues and concerns drive the applied software attribute measures and the overall analysis focus.

- The software attribute measures are not constrictive. Within the overall assessment structure, different measures can be applied for different programs. The integrity of the assessment structure with respect to a particular software development application ensures valid and objective analysis results. Measurement should be periodic and consistent. Program issues are readily identified as the quantitative data and associated attribute trends are updated.

- Specific measurement methodologies are defined and consistent across the development. The actual measures applied during the development change with respect to changing software development process activities and development products. Measurement is implemented at the lowest practical level consistent with the scope of the software development and the available resources. Multiple measures are applied with respect to a given issue to support cross-validation of the measurement findings.

- Software attribute measurement results are analyzed first on an attribute by attribute basis, comparing both expected (planned) and actual values. These measures and any associated findings are subsequently integrated within the assessment structure into an overall profile of the development processes and products. Multiple instantiations of "expected value" baselines are tracked for each attribute. Variances are measured from plan to plan and from plan to actual. Integration of the measurement results into an overall assessment is based upon software engineering principles, specific program characteristics, and historical attribute relationships.

- Software attribute measures are not interpreted as absolute determinants of the software management or technical issue in question. Software measures, although quantitative, are only indicators. As such, the assessment process calls for the use of multiple measures for a given issue, relating the results of the measures with established software engineering principals and relationships.

- The assessment process is supported by both commercial and locally developed automated measurement and assessment tools. These include data generation utilities, software development process models, metrics databases and utilities, attribute

assessment matrices, software product analyzers, graphics display interfaces, etc. These tools can be integrated into a software metrics analysis workstation which supports quantitative "what-if" analysis and the integration of the software metrics data into a high level software development assessment.

- **Within the software assessment structure, quantitative attribute data is combined with engineering observations to support validation of the assessment findings.**

- **In depth application of a "CORE" metrics set is preferable to the use of a large number of attribute measures. Such a "CORE" set of measures usually provides for the measurement of the following attributes at the software component, build, subsystem, and total program level:**

  - **Software Size**
  - **Software Development Effort / Resources**
  - **Software Development Schedule**
  - **Software Errors**

  These measures can be consistently applied prior to program initiation and through the traditional software development activities of requirements analysis, design, implementation, integration, and test.

SOFTWARE ISSUE DEFINITION

Software Issue Definition is the initial sub-process in the Quantitative Software Metrics Assessment Process. It is first implemented during the program planning phase prior to program implementation, and thereafter continues as the software development process proceeds and software products are designed, developed, tested, and released. The objective of Software Issue Definition, with respect to the overall assessment process, is to identify and prioritize the software process and product issues so that measurement and analysis efforts can be both focused and cost-effective.

Software Issue Definition encompasses issue identification, issue prioritization, and the mapping of issues to measurement requirements. Issues are initially defined based upon the technical, resource, and schedule characteristics of a particular software development program, and the constraints defined in the relationships between these characteristics. An aggressive development schedule, for example, impacts both the amount of capability which can be provided in the software functions as well as the development productivity and cost per unit of product. Severe resource constraints can materially impact overall software quality.

Figure 2 depicts some of the key areas of concern in military software development programs given the constraints of software development schedule, resources, capability, and development performance. This figure is an example of an issue definition structure which supports the Software Issue Definition sub-process.

SOFTWARE DEVELOPMENT ISSUES

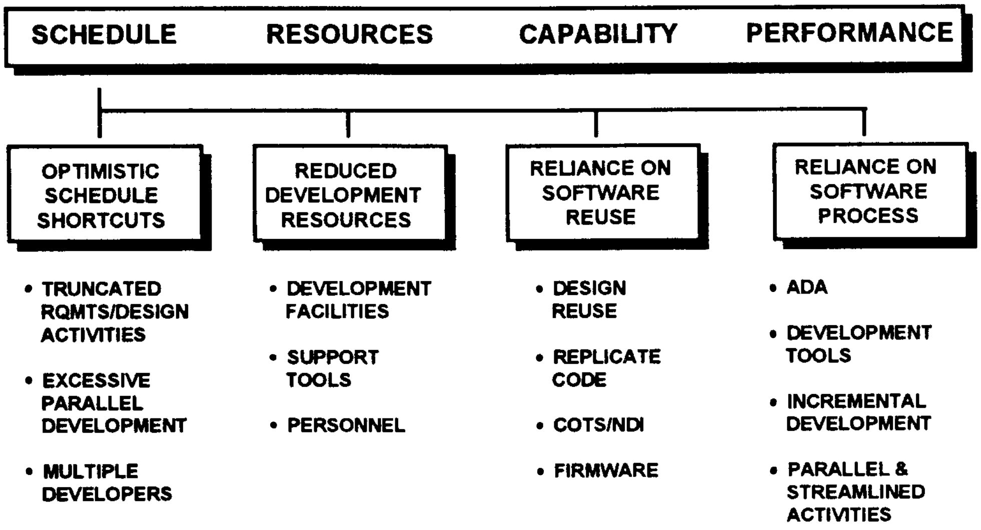
SCHEDULE
RESOURCES
CAPABILITY
PERFORMANCE
OPTIMISTIC SCHEDULE SHORTCUTS
REDUCED DEVELOPMENT RESOURCES
RELIANCE ON SOFTWARE REUSE
RELIANCE ON SOFTWARE PROCESS
• TRUNCATED RQMTS/DESIGN ACTIVITIES
• EXCESSIVE PARALLEL DEVELOPMENT
• MULTIPLE DEVELOPERS
• DEVELOPMENT FACILITIES
• SUPPORT TOOLS
• PERSONNEL
• DESIGN REUSE
• REPLICATE CODE
• COTS/NDI
• FIRMWARE
• ADA
• DEVELOPMENT TOOLS
• INCREMENTAL DEVELOPMENT
• PARALLEL & STREAMLINED ACTIVITIES

FIGURE 2

Initial development issues are continuously updated as the quantitative software metrics assessment process is iterated. New issues are identified and prioritized on a periodic basis. Software development issue definition and prioritization throughout the software development life cycle define the measurements which must be applied. Significant resource constraints, for example, require that the process attributes of software staffing, effort, facilities, cost performance, and productivity be measured and analyzed. Similarly, significant schedule constraints require the implementation of software progress and stability measures, as well as product completeness quality measures.

Figure 3 provides an example of commonly measured software attributes for a typical DOD software development program. Both process and product measures are represented.

SOFTWARE ATTRIBUTE MEASURES

PROCESS

- STAFFING
- EFFORT
- FACILITIES
- COST PERFORMANCE
- PRODUCTIVITY
- PROGRESS
  - MILESTONES
  - ACTIVITIES
  - FUNCTIONS
  - PRODUCTS
- STABILITY
- DEPENDENCIES
- STANDARDS CONFORMANCE

PRODUCT

- SIZE
- DEFECTS
- COMPLETENESS
- STABILITY
  - REQUIREMENTS
  - DESIGN
  - CODE
  - INTERFACES
  - PRODUCT ALLOCATIONS
- CONSISTENCY
- COMPLEXITY
- TRACEABILITY
- RELIABILITY
- EFFICIENCY

FIGURE 3

A number of tools and models support Software Issue Definition. These include the following:

- Software life cycle development and cost models (SLIM, COCOMO, SEER, etc.)
- Software sizing models
- Historical software development performance database
- Goal - Question - Metric Paradigm - (Basili, University of Maryland)
- Objectives - Principles - Attributes Paradigm - (Nance, Virginia Polytechnic Institute)
- Software Capability Maturity Model - (Software Engineering Institute)
- Software metrics historical databases
- Software quality metrics structures (language specific)

These tools and their underlying measurement and analysis structures are adapted to the defined metrics assessment structure and calibrated within the particular software development program. The functions provided by these analysis tools help to identify and prioritize the software development issues for a specific software development program. The output of these tools support the definition of specific attribute measures required for a given program.

SOFTWARE ATTRIBUTE MEASUREMENT

The objective of Software Attribute Measurement is to apply defined software measures to the software development processes and products. The set of measures which is applied is flexible, and is tailored to each specific program based upon identified development issues. The applied set also changes throughout the software development life cycle, as the measurable process and product attributes change with changing development activities.

A key requirement for Software Attribute Measurement is a well-defined measurement methodology for each attribute measure. This insures quantitative consistency within the program and allows for the actual measure to be tailored to specific program development processes. Consistency of the measurement methodology across programs is desirable for cross-program analysis, but is not mandatory for effective single program analysis. Although the overall assessment process is more focused upon the assessment of issues within a single development program, it can be extended to cross-program assessment based upon the implementation of consistent attribute measurement methodologies. This is usually a difficult task given the diverse scope of issues and characteristics which describe different development programs.

Measures of a given software attribute can take on many forms during the software development life cycle. Measures can represent estimates, plans, projections, process control limits, and actuals. A key part of the Software Attribute Measurement process is to ensure that each measure clearly reflects the process or product attribute which it represents, and that the measurement results are comparable across multiple forms. The way software lines of code (a size measure) is estimated, for example, must accurately represent the amount of functionality in the software, and must be comparable to actual lines of code measured when the software products are developed.

Software development attribute data is collected from numerous sources throughout the development process on a periodic basis. This data is cross-validated and analyzed and evaluated for availability, feasibility, and consistency. Attribute measures can be either automated or manual, and are defined with respect to specific software development processes and program characteristics. The ability to trace final assessment results and recommendations to the initial low level attribute measures is a significant part of the overall metrics assessment process

The primary tool for coordinating the Software Attribute Measurement process is a domain focused software metrics database. This database is the primary data processing and management tool, and is designed to support the characteristics and requirements of software measurement within the quantitative software measurement process. Such a database can be developed using a number of commercially available database engines and support applications. The software measurement schema design should provide for flexible adaptation of the database functions to different data sets and measurement methodologies for each unique software development program.

The software metrics database interfaces to numerous automated data entry tools and analysis applications. It forms the foundation for the software analysis workstation and related assessment tools and applications. Related tools and models pertinent to Software Attribute Measurement include the following:

- Metrics database and tools
- Software product attribute measurement tools (SLOC, Complexity, Language Constructs, etc.) - (ADAMAT, Complexity Analyzer, etc.)
- Developer and Government financial management systems (Effort, Cost)
- Project schedule planning and tracking systems (PERT, Artemis, etc.)
- Government validated Cost Schedule Control systems (DODI 7000.2)
- Software CASE tools - Usually program specific
- Software CM/Defect databases - Usually program specific

SOFTWARE INDICATOR GENERATION

The objective of Software Indicator Generation is to develop the quantitative reports and graphs which provide insight into the software development issues defined within the overall metrics assessment process. Based upon the measurement data collected in the Software Attribute Measurement sub-process, the indicators present information in a format which clarifies existing issues or helps to identify new issues. This is a key sub-process in that the data must be objective, but understandable at various management levels.

The actual software indicators which are produced within the quantitative software assessment structure vary with the applied attribute measurement set and with the level of analysis required to isolate the causes of data variances. For example, the variance between the planned amount of code to be produced over time and the actual amount produced may be not be apparent at the software system level. When broken down into its component parts, however, the lower level indicators generated for each software functional component may show that one component has a positive variance and one a negative variance and that these offset each other when the indicator is aggregated at the system level. A key part of the quantitative software metrics assessment structure is the use of low level attribute data which can be used to generate different indicators during the analysis process as dictated by changing analysis questions and new hypothesis. Figure 4 is an example of a software indicator. It depicts a single software attribute, software size expressed in Source Lines Of Code (SLOC), over time. Both planning and actual data is represented.

SOFTWARE SIZE INDICATOR

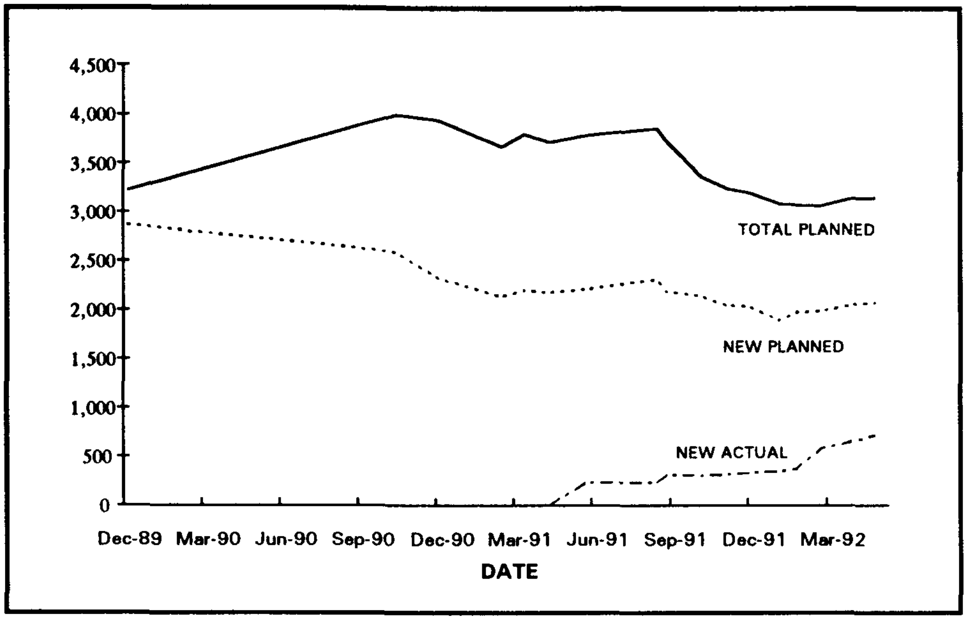

FIGURE 4

This is an example of an indicator which is based upon only a single attribute. Based upon analysis requirements many types of indicators can be generated, including those which incorporate multiple attribute types. The indicator is valid if it materially represents the processes and products of the software development program.

Within the quantitative assessment structure, indicators which address common issues or which are generated consistently across the development life cycle are considered to be "CORE indicators" and are derived from the "CORE metrics" attribute data. These are generated on a periodic basis and are augmented by other indicators deemed necessary to support specific analysis questions and requirements.

The techniques and products of the Software Indicator Generation process build upon those of Software Attribute Measurement to provide a basis for integrated software quantitative analysis and assessment.

Tools and models which support Software Indicator Generation include the following:

- Software metrics database
- Software metrics analysis workstation
- Commercial graphics and reporting applications

SOFTWARE QUANTITATIVE ASSESSMENT

Software Quantitative Assessment is a key sub-process in the overall software metrics assessment process. The objectives of this process are as follows:

- To clarify quantitatively the existing software development issues,
- To identify new or possible software development issues,
- To evaluate the degree of impact of a given software development issue,
- To correlate process and product analysis issues, and
- To generate recommendations for improvement.

The Software Quantitative Assessment process incorporates the findings, products, and results from other software assessment sub-processes and integrates them into an overall objective profile of the software development program. This is accomplished by relating all quantitative findings within a context of software engineering principles and specific program characteristics and engineering observations.

A key focus of Software Quantitative Assessment is the isolation of those components of the software development which demonstrate the highest degree of development cost, schedule, or technical risk. This is accomplished by applying multiple measures and indicators to low level software components, and then comparing the components based upon the resultant findings. When this technique is applied over time, key risk areas can be identified and addressed. An analysis tool used for this purpose is an evaluation matrix which displays multiple attribute measures for the major software components of a current program. When generated on a periodic basis, the matrix can provide the basis for the generation of risk profiles and trend indicators for any aggregation of the software design structure.

Another key focus is the analysis and determination of cause and effect relationships between the software development process and the resultant software products. Experience has shown that development constraints in terms of resources, schedule, and process capability on the part of the developer can significantly impact the integrity, efficiency, and adequacy of the software development process as implemented. These process deficiencies subsequently impact the amount of capability and quality of the related software products. The Software Quantitative Assessment process relates measures of software process and product attributes through the implementation of a sequential analysis approach. Process measures are assessed early in the development, and indicators are generated which focus on identified constraints. As the attribute data becomes available, product indicators are generated and correlated to the process results for the defined software components. In this manner the structure supports the projection of possible product shortfalls based upon process indicators. This approach is depicted in Figure 5.

PROCESS - PRODUCT ANALYSIS SEQUENCE

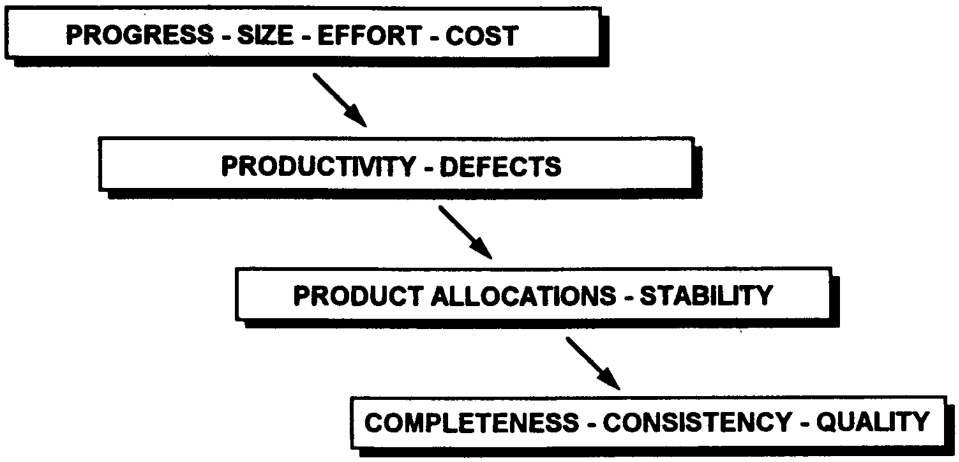
PROGRESS - SIZE - EFFORT - COST
PRODUCTIVITY - DEFECTS
PRODUCT ALLOCATIONS - STABILITY
COMPLETENESS - CONSISTENCY - QUALITY

FIGURE 5

Software Quantitative Assessment also incorporates the integration of subjective engineering analysis data with the quantitative data derived from the implementation and analysis of software metrics. The correlation of these two types of data provides a methodology for cross-checking the software analysis results.

All tools and models previously identified are implemented to support the Software Quantitative Assessment process. In addition, the following resources are also applied:

- Statistical Analysis models and application packages
- Spreadsheet/Graphics application packages
- Historical software attribute databases

CONCLUSION

The definition and implementation of a structured software measurement process has been the basis for applied software metrics on a number of DOD software development programs. The process is flexible, and supports the use of different measures to assess distinct issues on programs with differing characteristics. The process is consistent, however, in that the measurement activities are documented and continuous across a particular software development program. Both software development process and product related measures are applied within the measurement process, and the measurement results are integrated with other program information to comprise an overall software development evaluation.