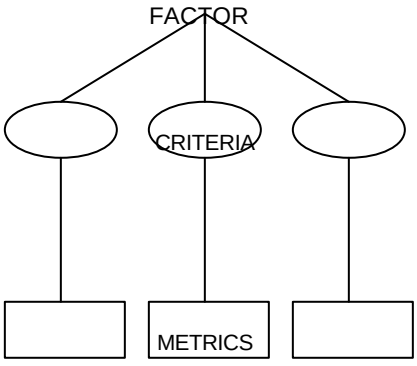
MEASUREMENT FRAMEWORKS
II

User's View of Quality
~~Factor/Criteria/Metric Paradigm~~



Management Oriented View of Product Quality

(CORRECTNESS)

Software Oriented Attributes which provide quality

(COMPLETENESS)

Quantitative Measures of those attributes

(UNAMBIGUOUS REFERENCES)

- product characteristics
- customer point of view

Boehm, Thayer, Lipow (TRW)

McCall, Walters (GE)

Boeing Harris

Software Quality Metrics

Performance

How well does it Function?

User Concern	Quality Factor
How well does it utilize a resource?	Efficiency
How secure is it?	Integrity
What confidence can be placed in what it does?	Reliability
How well will it perform under adverse conditions?	Survivability
How easy is it to use?	Usability

Software Quality Metrics

Design

How valid is the Design?

User Concern	Quality Factor
How well does it conform to the requirements?	Correctness
How easy is it to repair?	Maintainability
How easy is it to verify its performance?	Verifiability

Software Quality Metrics

Adaptation

How adaptable is it?

User Concern	Quality Factor
How easy is it to expand or upgrade its capability or performance?	Expandability
How easy is it to change?	Flexibility
How easy is it to interface with another system?	Interoperability
How easy is it to transport?	Portability
How easy is it to convert for use in another application?	Reusability

Software Quality Metrics

Evaluating Factors

Based upon project needs, various quality factors are chosen for monitoring, analysis, and feedback for corrective action.

Each Factor is defined by a rating formula whose data is not available till project completion

Each Factor is also defined by a set of criteria that are collected during project development and are used as predictors of the quality factors

These metrics are collected on worksheets

Software Quality Metrics

Quality Factor Formulas

Efficiency	1 -	$\frac{\text{Actual Utilization}}{\text{Allocated Utilization}}$
Integrity	1 -	$\frac{\text{Errors}}{\text{Lines of Code}}$
Reliability	1 -	$\frac{\text{Errors}}{\text{Lines of Code}}$
Survivability	1 -	$\frac{\text{Errors}}{\text{Lines of Code}}$
Usability	1 -	$\frac{\text{Labor-Days to Use}}{\text{Labor-Years to develop}}$
Correctness	1 -	$\frac{\text{Errors}}{\text{Lines of Code}}$
Maintainability	1 -	0.1(Average Labor-Days to Fix)

Software Quality Metrics

Quality Factor Formulas

Verifiability	1 -	$\frac{\text{Effort to Verify}}{\text{Effort to Develop}}$
Expandability	1 -	$\frac{\text{Effort to Expand}}{\text{Effort to Develop}}$
Flexibility	1 -	0.05 (Average Labor-Days to Change)
Interoperability	1 -	$\frac{\text{Effort to Couple}}{\text{Effort to Develop}}$
Portability	1 -	$\frac{\text{Effort to Transport}}{\text{Effort to Develop}}$
Reusability	1 -	$\frac{\text{Effort to Convert}}{\text{Effort to Develop}}$

Software Quality Metrics

Quality Criteria

Each factor has associated with it a set of quality criteria:

Factor	Criteria
Efficiency	Effectiveness-Communication
	Effectiveness-Processing
	Effectiveness-Storage
Integrity	System Accessibility
Reliability	Accuracy
	Anomaly Management
	Simplicity
Survivability	Anomaly Management
	Autonomy
	Distributedness
	Modularity
	Reconfigurability

Software Quality Metrics

Quality Criteria

Factor	Criteria
Usability	Operability
	Training
Correctness	Completeness
	Consistency
	Traceability
Maintainability	Consistency
	Modularity
	Self-Descriptiveness
	Simplicity
	Visibility
Verifiability	Modularity
	Self-Descriptiveness
	Simplicity
	Visibility

Software Quality Metrics

Quality Criteria

Factor	Criteria
Expandability	Augmentability
	Generality
	Modularity
	Self-Descriptiveness
	Simplicity
	Virtuality
Flexibility	Generality
	Modularity
	Self-Descriptiveness
	Simplicity
Interoperability	Commonality
	Functional Overlap
	Independence
	Modularity
	System Compatibility

Software Quality Metrics

Quality Criteria

Factor	Criteria
Portability	Independence
	Modularity
	Self-Descriptiveness
Reusability	Application Independence
	Document Accessibility
	Functional Scope
	Generality
	Independence
	Modularity
	Self-Descriptiveness
	Simplicity
	System Clarity

Software Quality Metrics

Metrics

Each criteria has associated with it a set of quality metrics:

Example:

Criteria	Metrics
Completeness	CP 1: Completeness Checklist
Consistency	CS 1: Procedure Consistency CS 2: Data Consistency
Traceability	TR 1: Cross Reference

Software Quality Metrics

Worksheets

Associated with each phase of development are worksheets that collect data that can be turned into quality metrics

For Example:

Metric Worksheet 0:	Collected during Systems Requirements Analysis
Metric Worksheet 1:	Collected during Software Requirements Analysis
Metric Worksheet 2:	Collected during Preliminary Design
Metric Worksheet 3:	Collected during Detailed Design
Metric Worksheet 4:	Collected during Coding and Unit Test

Software Quality Metrics

Evaluation

Data collected at the unit level is accumulated at the subsystem level

Data collected at the subsystem level is accumulated at the system level

Accumulation can be done by averaging the values of metrics at the lower level units

Score sheets exist for each factor

Metric values at the system level are reported for each criteria

All metric values are designed to lie between 0 and 1.

Software Quality Metrics

Metric Definition

Example: Completeness
From Worksheet 0:

CP.1(1) Has a common technical vocabulary with equivalent definition been established for use with this system and for use with the interoperating systems (e.g., definition and use of data item, block, record)? Y, N, N/A

CP.1(2) a. How many data references are identified? ___, N/A
b. How many identified data references are documented with regard to source, meaning, and format? ___, N/A
c. Calculate b/a and enter score. ___, N/A

CP.1(3) a. How many data items are defined (i.e., documented with regard to source, meaning and format)? ___, N/A
b. How many data items are referenced? ___, N/A
c. Calculate b/a and enter score. ___, N/A

CP.1(5) Have all defined functions been referenced? Y, N, N/A

Software Quality Metrics

Metric Definition

CP.1(6) Have all system functions been allocated to configuration items? Y, N, N/A

CP.1(7) Have all referenced functions been defined (i.e., documented with precise inputs, processing, and output requirements)? Y, N, N/A

CP.1(8) Are the processing flows (algorithms) and all decision points (conditions and alternate paths) in the flows described for all functions? Y, N, N/A

CP.1 (11) a. How many software problem reports have been recorded to date? ___, N/A

b. How many recorded software problem reports have been closed to date? ___, N/A

c. Calculate b/a and enter score ___, N/A

Software Quality Metrics

Metric Definition

Example: Completeness - Other Worksheets

Worksheet 1: Same as Worksheet 0 except for CP.1(1)

CP.1(1) All all inputs, processing and outputs clearly defined? Y, N, N/A

Worksheet 2: Same as Worksheet 1 except CP.1(5,7,8) are eliminated and CP.1(9) is added

CP.1(9) Are all conditions and alternative processing options defined for each decision point? Y, N, N/A

Worksheet 3B: Same as Worksheet 2 except CP.1(3,6,7,8) are eliminated and CP.1(10) is added

CP.1(10) Are all parameters in the argument list used? Y, N, N/A

Software Quality Metrics

Metric Definition

Example: Completeness - Other Worksheets

Worksheet 3A tends to summarize the results of Worksheet 3B,
i.e., CP.1(1,2,3,4,9,10,11) are summed as follows:

CP.1(1) a.How many applicable units (answer of Y or N on 3B)? ___, N/A
b.How many units with answer of Y on 3B? ___, N/A
c. Calculate b/a and enter score. ___, N/A

Worksheet 4B has CP.1(2,4,9,10)

Worksheet 4A has CP.1(2,3,4,9,10,11)

Software Quality Metrics

Relationship between SQM and GQM

We can express the SQM in the GQM framework by writing them as goals for the following form:

Analyze the **final product** for the purpose of **assessment** with respect to **efficiency** from the point of view of the **customer**.

Analyze the **final product** for the purpose of **assessment** with respect to **integrity** from the point of view of the **customer**.

That is, they all attempt to assess the final product from the point of view of the customer relative to the various quality factors

Analyze the **final product** for the purpose of **assessment** with respect to **efficiency, integrity, reliability, survivability, usability, correctness, maintainability, verifiability, expandability, flexibility, interoperability, portability, reusability** from the point of view of the **customer**.

Comparing Quality Frameworks

Criteria	QFD Approach	SQM Approach	GQM Approach
Scope of Goals			
Objects	products	final product	any
Purpose	plan (engineer, control)	certify	any
Viewpoint	customer,user	customer, user	any
Identification of Metrics			
Paradigm	Trace user characteristics of final product into related product/process characteristics at various stages of development	Refine factors into criteria and metrics	Refine goals into questions and metrics
Mechanism	UC: select/tailor PC: select/tailor	F: select C: select M: select	G: select/tailor Q: select/tailor M: select/tailor
Scope of the Approach			
Use	Quality Management	Quality Management	Quality and Project Management

References

V. Basili and D. Weiss, A Methodology for Collecting Valid Software Engineering Data, IEEE Transactions on Software Engineering, pp. 728-738, November 1984.

D. Weiss and V. Basili, Evaluating Software Development by Analysis of Changes: The Data from the Software Engineering Laboratory, IEEE Transactions on Software Engineering, pp. 157-168. February 1985.

V. Basili, Viewing Maintenance as Reuse-Oriented Software Development, IEEE Software, pp. 19-15, January 1990.

V. Basili and D. Weiss, Evaluation of the A-7 Requirement s Document by Analysis of Change Date, Proceedings of the Fifth International Conference on Software Engineering, pp. 314-323, March 1981.

V. Basili, Applying the Goal/Question/Metric Paradigm in the Experience Factory, 10th Annual CSR (Centre for Software Reliability) Workshop, Application of Software Metrics and Quality Assurance in Industry, Amsterdam, Holland, 29th September - 1st October, 1993.