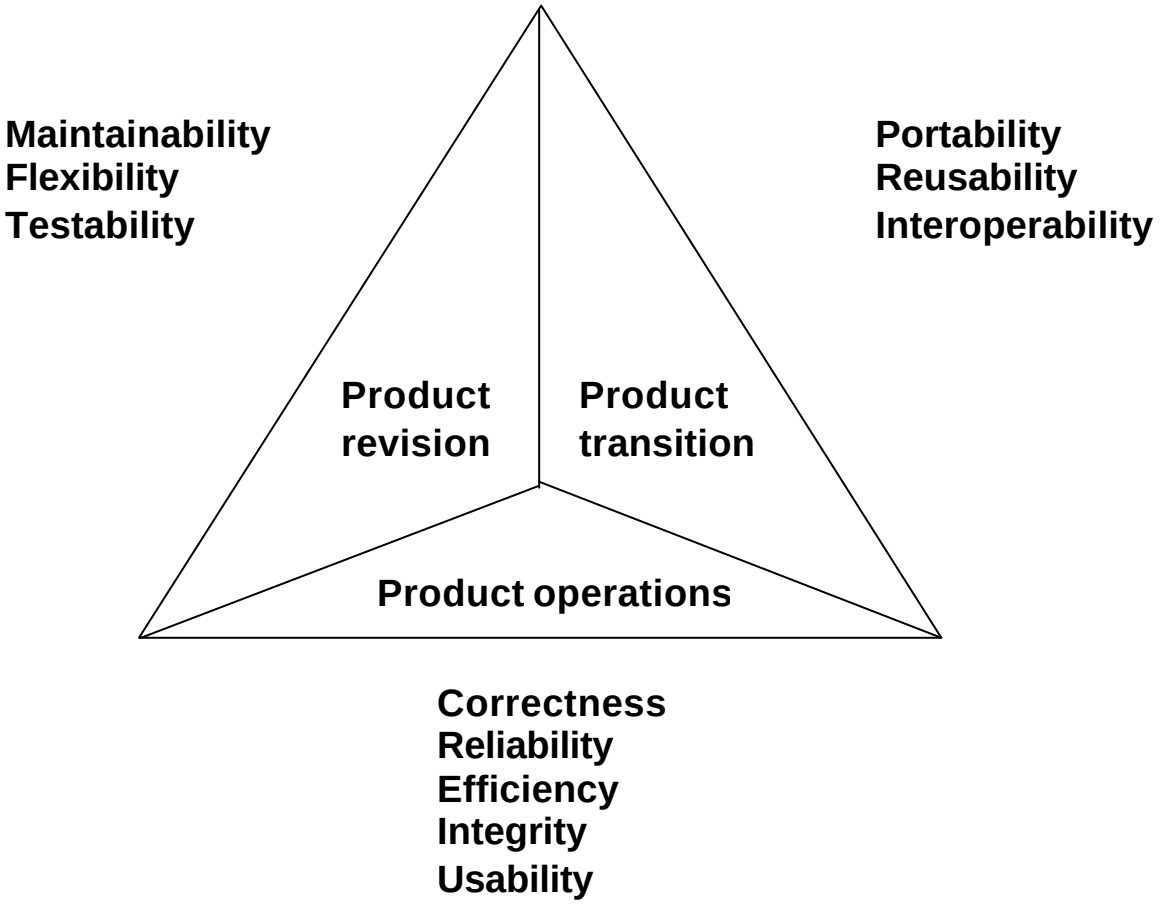
McCall's Software Quality Checklist

McCall's Software Quality Factors (from Lecture 10A)



It is difficult, and in some cases impossible, to develop direct measures of the above quality factors. McCall's approach is to define a set of metrics and develop expressions for each quality factor according to:

$$F_q = c_1 m_1 + c_2 m_2 + ... + c_n m_n$$

where F_q is a quality factor, c_n are regression coefficients and m_n are the metrics that affect that quality factor. Unfortunately, many of the metrics McCall defined for these purposes can only be defined subjectively. The metrics may be used in the form of a checklist used to grade subjectively specific aspects of the software. McCall's proposed metrics, to be graded on a scale from 0 (low) to 10 (high), are shown in

Table 1 McCall's metrics contributing to software quality factors

Auditability	*The ease with which conformance to standards can be checked*
Accuracy	*The precision of computations and control*
Communication commonality	*The degree to which standard interfaces, protocols and bandwidth are used*
Completeness	*The degree to which full implementation of the required function has been achieved*
Conciseness	*The compactness of the program in terms of lines of code*
Consistency	*The use of uniform design and documentation techniques throughout the software development protocol*
Data commonality	*The use of standard data structures and types throughout the program*
Error tolerance	*The damage that occurs when a program encounters an error*
Execution efficiency	*The run-time performance of the program*
Expandability	*The degree to which architectural, data or procedural design can be extended*
Generality	*The breadth of potential application of program components*
Hardware independence	*The degree to which the software is decoupled from the hardware on which it operates*
Instrumentation	*The degree to which the program monitors its own operations and identifies errors that do occur*
Modularity	*The functional independence of program components*
Operability	*The ease of operation of the program*
Security	*The availability of mechanisms that control or protect programs and data*
Self-documentation	*The degree to which the source code provides meaningful documentation*
Simplicity	*The degree to which a program can be understood without difficulty*
Software system independence	*The degree to which the program is independent of non-standard programming language features, operating system characteristics and other environmental constraints*
Traceability	*The ability to trace a design representation or actual program component back to requirements*
Training	*The degree to which the software assists in enabling new users to apply the system*

Each of these metrics contributes to one or more quality factors, as shown in Table 2. Note that the weight given to each metric depends on individual products and concerns.

Table 2 Relationships between McCall's quality factors and metrics

Quality factor / Software quality metric	Correctness	Reliability	Efficiency	Integrity	Maintainability	Flexibility	Testability	Portability	Reusability	Interoperability	Usability
Auditability				X			X				
Accuracy		X									
Communication commonality										X	
Completeness	X										
Conciseness			X		X	X					
Consistency	X	X			X	X					
Data commonality										X	
Error tolerance		X									
Execution efficiency			X								
Expandability						X					
Generality						X		X	X	X	
Hardware independence								X	X		
Instrumentation				X	X		X				
Modularity		X			X	X	X	X	X	X	
Operability			X								X
Security				X							
Self-documentation					X	X	X	X	X		
Simplicity		X			X	X	X				
Software system independence								X	X		
Traceability	X										
Training											X

References

This material is adapted from:

Roger S. Pressman, "Software Engineering: A Practitioner's Approach" (European Adaptation), Ch. 19, Fifth Edition, 2000.
(note that earlier editions also contain this material)