

# Defining Quality Workbook

**<Program/Project/Work Name>  
Quality Definition**

# Introduction: Defining Quality

When starting on a piece of work it is important to understand what you are working towards. Much of the time, scope / time / cost has value attached to it where quality is ambiguous with multiple definitions.

This Defining Quality pack has been put together to bring the team to a common understanding on the definition of Quality.

This pack consists of instructions for team members on how to define Quality followed by a template where users can work through the steps below to create their own quality definition.



# Overview: Quality Definition

Quality is a multi-dimensional **measure that describes how a system or application satisfies the stated or implied requirements**. Stated requirements are equivalent to functions or features and implied requirements are equivalent to performance, useability, security, maintainability or other non-functional requirements.

For projects, **quality must be agreed and continually reviewed to ensure all the requirements of stakeholders (stated or implied) are properly satisfied**. Software development projects often focus too heavily on satisfying functional requirements and overlook requirements for maintainability, flexibility and performance.

**Everyone in the team must have the same understanding of quality**, how it is measured and the impact that has on their working practices. If this agreement is not reached, then the process for ensuring quality is maintained can not be agreed upon.

*Everyone is Responsible for Quality!*



# 1 Quality Advocates

Name: Peter Gibbons  
Role: Software Engineer

Quality means: Low level of technical debt, good unit test coverage and a green CI build

Name: Temperance Brennan  
Role: Test Engineer

Quality means: The system works from end to end and we have confidence in deploying.

Name: Michael Scott  
Role: Project Manager

Quality means: Low level of defects and on-time delivery

Name:  
Role:

Quality means:

## **INSTRUCTIONS**

*Identify the quality advocates in your team and what quality means to each role / skillset. Ensure you have representation from the business, technical, testing and management.*

*This works well as a sticky note session, where everybody writes their definition and shares it with the rest of the team.*

*Delete this instruction and replace the examples above with the definitions from your entire team.*

*Add new pages as required.*

# Overview: Quality Taxonomy



Quality Attribute	Meaning
Correctness	The degree to which the features or functions need to be correct
Reliability	The probability of the system operating without failure for a certain period of time
Robustness	The degree to which the system functions correctly when confronted with invalid input data, software or hardware defects, or unexpected operating conditions
Integrity	Ability to preclude unauthorised access, prevent information loss, protect from viruses infection, protect privacy of data entered
Efficiency	How well the system uses available resources – processor capacity, disk space, memory, communication bandwidth
Portability	The effort required to migrate a piece of software from one operating environment to another
Reusability	The extent to which a software component can be used in other applications
Interoperability	Ease with which the system can exchange information with other systems
Maintainability	How easy it is to correct a defect or make a change in the system
Flexibility	How much effort is required to add more capabilities to the product
Testability	The ease of which the components of the system can be tested to find defects
Installability	The ease of installation of the software
Availability	Percentage of planned 'uptime' that the system is required to be operational
Survivability	The amount of time the system is required for use
Usability	Measures the effort required to prepare input for, operate, and interpret the output of the product

# Overview: Quality Attribute Tradeoffs



If you have...	...you get...	Reliability	Robustness	Availability	Integrity	Flexibility	Usability	Interoperability	Efficiency	Testability	Maintainability	Reusability	Portability
Reliability			+	+		+	+		-	+	+		
Robustness	+		+				+		-				
Availability	+	+											
Integrity							-	-	-	-		-	
Flexibility	+				-				-	+	+		+
Usability			+						-	-			
Interoperability					-	+			-				+
Efficiency	-	-				-	-	-		-	-		-
Testability	+		+			+	+		-		+		
Maintainability	+		+			+			-	+			
Reusability	-				-	+		+	-	+	+		+
Portability						+	-	+	-	+	-	+	

Priority	Quality Attribute	Meaning
5	Reliability	The probability of the system operating without failure for a certain period of time
3	Robustness	The degree to which the system functions correctly when confronted with invalid input data, software or hardware defects, or unexpected operating conditions
2	Integrity	Ability to preclude unauthorised access, prevent information loss, protect from viruses infection, protect privacy of data entered
	Efficiency	How well the system uses available resources
	Portability	The effort required to migrate a piece of software from one operating environment to another
2	Reusability	The extent to which a software component can be used in other applications
1	Interoperability	Ease with which the system can exchange information with other systems
10	Maintainability	How easy it is to correct a defect or make a change in the system
4	Flexibility	How much effort is required to add more capabilities to the product
4	Testability	The ease of which the components of the system can be tested to find defects
	Installability	The ease of installation of the software

### INSTRUCTIONS

*The idea here is to define the key quality attributes for the team, based on the wisdom of the crowd.*

*Correctness is a given for any piece of work, so don't include this attribute.*

*Prioritise your quality attributes. You could do this via [MoSCoW technique](#) or by getting all of the attendees to nominate their top 3 attributes via voting.*

*Delete this instruction and replace the example above.*

Attribute	Meaning
Usability	<ul style="list-style-type: none"> <li>• Can people use it, less training, easy to learn</li> <li>• Logical, easy to use, intuitive</li> <li>• Completeness – quote and application</li> <li>• Efficiency</li> </ul>
Maintainability	<ul style="list-style-type: none"> <li>• Easy to change, speed to market</li> <li>• Simple</li> <li>• Fast to improve, responsiveness</li> <li>• TCO, life of product</li> </ul>
Reliability	<ul style="list-style-type: none"> <li>• Consistent availability</li> <li>• No crashes, gracefully crashes</li> <li>• Consistent outcome</li> <li>• Robustness</li> </ul>

#### INSTRUCTIONS

*We are now looking for our top 5 quality attributes and OUR meaning of them. Reducing the quality attributes from the quality taxonomy allows the ability for the team to focus on what is meaningful and important to their piece of work.*

*First, take the top 4 attributes from step 2 and add them to this table.*

*Then, as a group, agree what these attributes mean to us as a team.*

*We will add the fifth attribute after the next step.*

*Delete this instruction and replace the example above.*

# Quality Taxonomy – Tradeoff Risks



Quality Attribute	Tradeoff Risk
Maintainability	Unable to easily upgrade the system Wasted time on maintenance
Reliability	Loss of income
Useability	Unhappy users
Integrity	Security breaches Allowing unauthorised access
Reliability	System outage
Flexibility	Overly complex system
Portability	Unable to use multiple platforms
Interoperability	Unable to interact with other systems
Testability	Insufficient testing capability
Survivability	Only has a short shelf life
Robustness	Error prone
Installability	Unable to install software on local machines

**Low Quality = Technical Debt**

Priority	Quality Attribute	Risk
6	Flexibility	<ul style="list-style-type: none"><li>• Covered in maintainability?</li><li>• Hard to meet time to market, miss some scope</li><li>• Less competitive</li></ul>
4	Testability	<ul style="list-style-type: none"><li>• Errors not found, low confidence</li><li>• Reliability impact, cost of change</li><li>• More effort and time</li><li>• Resource strain</li><li>• Complex dependencies</li></ul>
5	Robustness	<ul style="list-style-type: none"><li>• Low customer confidence</li><li>• Increase training, support costs</li><li>• Higher cost of change, manual workaround</li><li>• Brand impact, complaints</li></ul>

**INSTRUCTIONS**

*We now deal with the remaining quality attributes and decide OUR risk of not completing them.*

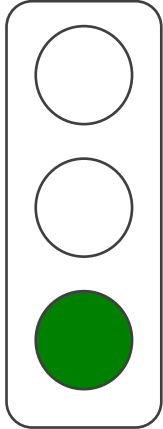
*First, add the quality attributes from step 2 that we did not prioritise in step 3.*

*Then, as a group, agree what the risk is of not meeting this attributes as a team.*

*Next, prioritise your quality attributes. You could do this via [MoSCoW technique](#) or by getting all of the attendees to nominate their top risk via voting. Take the top priority risk and add it to the attributes in step 2.*

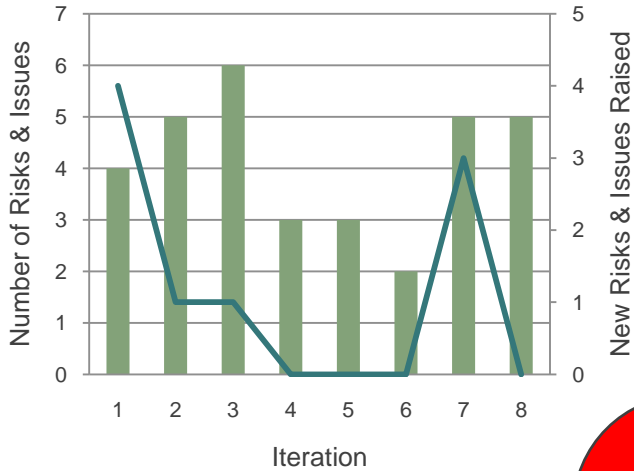
*Delete this instruction and replace the example above.*

# Overview: Quality Measurement

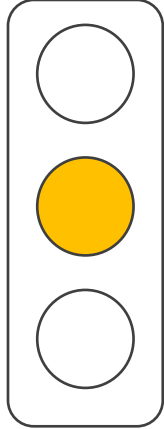
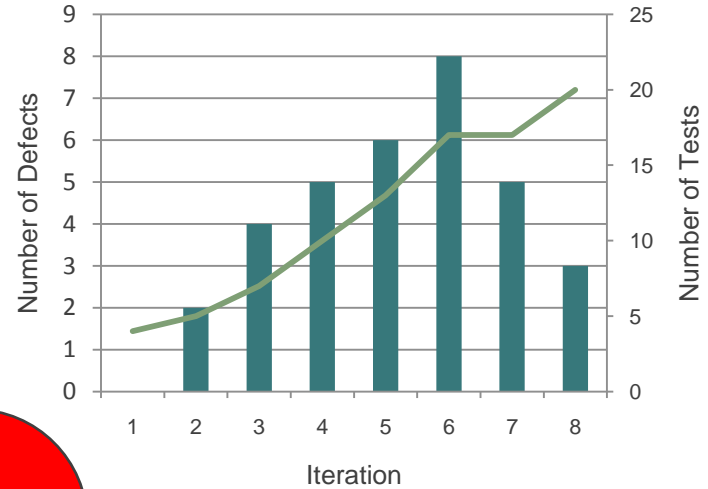


HEALTH

## PROJECT



## TESTING

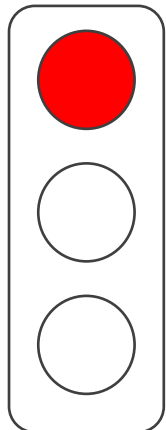
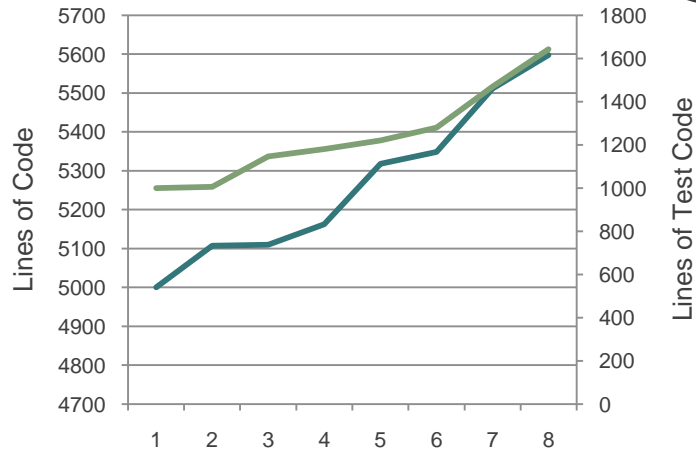


TEST COVERAGE

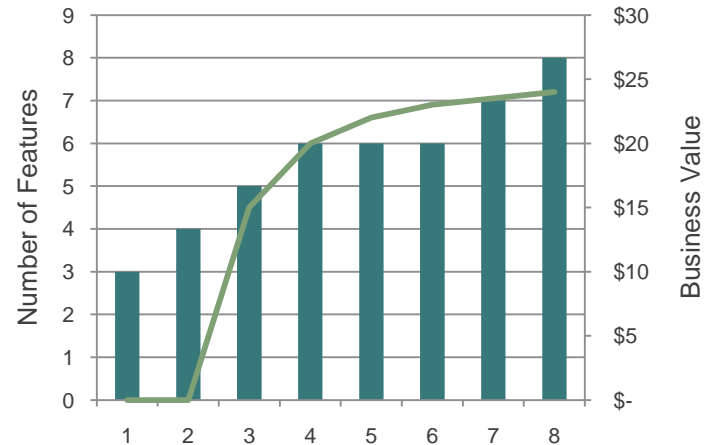


OVERALL

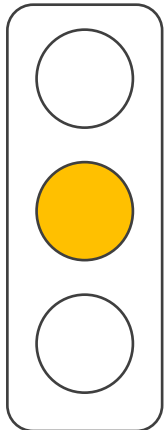
## MAINTAINABILITY DEVELOPMENT



## USER



## PERFORMANCE



Attribute	Test Criteria	SMART* Goal / Metric
Efficiency	Execution Efficiency	X% of the available processor capacity shall be unused at peak load conditions (defined elsewhere)
Reliability	Simplicity	The McCabe Complexity of all modules shall be below 10; the average of all modules shall be below 5
Availability	Mean Time to Repair	The average time required to bring the system back online after failure shall not exceed 10 minutes
Usability	Response Times	The average time required to generate and display an online report shall be less than 2 seconds, and no online reports shall take more than 5 seconds. For those that require more than 250 milliseconds, there shall be graphical feedback to the user that the system is processing data

### INSTRUCTIONS

*We now want to discuss how we measure that we are meeting our quality attributes. Adding Test Criteria to the Quality Attributes allows the team to understand how the Quality Attribute will be tested and assigns it a definition of done. Adding SMART goals enables the parameter around the attributes test criteria to be set and outlines definition on how to achieve the test criteria.*

*First, transfer the final list of quality attributes from step 3.*

*Then, determine the test criteria and then the SMART goals and specific metrics to measure each attribute.*

*Delete this instruction and replace the example above.*

*THIS STEP CAN BE DEFERRED TO A SEPARATE DISCUSSION IF TIME IS NOT AVAILABLE.*

# Overview: Quality Success Slider Meaning



Quality Slider Level	Explanation
1	Work your hardest to sign off all your quality attributes and associated testing criteria – application is critical, impact on business is high, media will become involved if issues arise
2	Look at signing off your top three quality attributes and their associated testing criteria – core piece of infrastructure, future development will be undertaken and the need to expand or change quickly is high
3	Pick one or two of your quality attributes (focussing on the ones that give most value to the work) and aim at completing those – other pressures means quality needs to be sacrificed a high level of technical debt may be incurred
4	Do your best to undertake some of the testing criteria, focussing on the most critical in the time you have (this doesn't mean not doing anything) – you get what you get which will be basic correctness, a high level of technical debt will be incurred

**A discussion needs to be had and agreement needs to be made on what it means if the quality slider is set at a certain level.**

**If the quality slider is not number one then some form of technical debt will be accrued.**

**A risk needs to be raised and a strategy put in place to address this.**

## SUCCESS TYPE

## SUPPORTING EXPLANATION

## SCOPE – meet the projects requirements



New and existing records converted, all reports available

## COST – meet agreed budget



Budget = \$500K

## TIME – deliver the project on time



Due by June

## QUALITY – meet quality requirements



No high priority bugs, performance levels met, ability to add new functions easily in next release

**INSTRUCTIONS**

Now either define your [success sliders](#) OR refer to the success sliders that have already been defined for your piece of work.

Only one slider can sit at each level. Set the position of the sliders to reflect your understanding of how the success of this project will be judged when it's finished.

Review the position of the quality slider against the meaning on the previous page and discuss amongst the team if we are comfortable with this. If you are not, you will need to adjust the sliders.

Delete this instruction and replace the example above.

**As a team we define quality as** *<summary of what will make up your definition of quality using output from step 3, based on the position of the success slider in step 6>*

**We will strive to achieve quality by** *<summary of the test criteria or SMART goals from step 5, based on the position of the success slider in step 6 >*

**If <time, cost, scope > permits we will focus additionally on** *<add the quality attributes listed in step 3 and/or step 4, based on the position of the success slider in step 6 (only add in if quality is not the number one slider)>*

**Due to project constraints we believe we will not be able undertake** *<add the quality attributes that the team believes will not be able to be completed using output from step 4, along with a brief summary of the major risks>*

## INSTRUCTIONS

*The quality definition is our clear statement of what quality means (what we will and will not focus on). This does not necessarily mean we will not do something about all of our quality attributes, just that some will not be our focus. This statement should be placed in a prominent area with the team.*

*Add a summary based on the inline instructions above.*

*Delete this instruction and replace the example above.*

# Overview: Next Steps (Quality Practices)



Agile Quality Practices							AGILE ACADEMY		
The worksheet is designed to help teams assess the maturity of their practices and to identify opportunities for improvement. It is not intended for comparison or scoring of teams.									
TEAM:		DATE:		INSTRUCTIONS:		GLOSSARY:			
OBSERVATIONS:		RECOMMENDATIONS:							
<ul style="list-style-type: none"> <li>•</li> <li>•</li> <li>•</li> </ul>		<ul style="list-style-type: none"> <li>•</li> <li>•</li> <li>•</li> </ul>				<ul style="list-style-type: none"> <li>• The worksheet should be completed by whole teams to ensure consistency within teams.</li> <li>• Each team should assess the listed quality practices and assign a maturity level to each.</li> </ul>			
Level	-1	0	1	2	3	4	5	Observations	Recommendations
Requirements	1. Requirements are captured using a formal template managed with version control.	1. Requirements are captured using a formal template managed with version control.	1. Requirements are captured using a formal template managed with version control.	1. Requirements are captured using a formal template managed with version control.	1. Requirements are captured using a formal template managed with version control.	1. Requirements are captured using a formal template managed with version control.	1. Requirements are captured using a formal template managed with version control.		
	2. Requirements are captured using a formal template managed with version control.	2. Requirements are captured using a formal template managed with version control.	2. Requirements are captured using a formal template managed with version control.	2. Requirements are captured using a formal template managed with version control.	2. Requirements are captured using a formal template managed with version control.	2. Requirements are captured using a formal template managed with version control.	2. Requirements are captured using a formal template managed with version control.		
Development	1. Code is developed in a structured manner with clear ownership and accountability.	1. Code is developed in a structured manner with clear ownership and accountability.	1. Code is developed in a structured manner with clear ownership and accountability.	1. Code is developed in a structured manner with clear ownership and accountability.	1. Code is developed in a structured manner with clear ownership and accountability.	1. Code is developed in a structured manner with clear ownership and accountability.	1. Code is developed in a structured manner with clear ownership and accountability.		
	2. Code is developed in a structured manner with clear ownership and accountability.	2. Code is developed in a structured manner with clear ownership and accountability.	2. Code is developed in a structured manner with clear ownership and accountability.	2. Code is developed in a structured manner with clear ownership and accountability.	2. Code is developed in a structured manner with clear ownership and accountability.	2. Code is developed in a structured manner with clear ownership and accountability.	2. Code is developed in a structured manner with clear ownership and accountability.		
TEST	1. Test cases are developed and executed in a structured manner.	1. Test cases are developed and executed in a structured manner.	1. Test cases are developed and executed in a structured manner.	1. Test cases are developed and executed in a structured manner.	1. Test cases are developed and executed in a structured manner.	1. Test cases are developed and executed in a structured manner.	1. Test cases are developed and executed in a structured manner.		
	2. Test cases are developed and executed in a structured manner.	2. Test cases are developed and executed in a structured manner.	2. Test cases are developed and executed in a structured manner.	2. Test cases are developed and executed in a structured manner.	2. Test cases are developed and executed in a structured manner.	2. Test cases are developed and executed in a structured manner.	2. Test cases are developed and executed in a structured manner.		
TEAM	1. Team members are actively engaged in the team's work.	1. Team members are actively engaged in the team's work.	1. Team members are actively engaged in the team's work.	1. Team members are actively engaged in the team's work.	1. Team members are actively engaged in the team's work.	1. Team members are actively engaged in the team's work.	1. Team members are actively engaged in the team's work.		
	2. Team members are actively engaged in the team's work.	2. Team members are actively engaged in the team's work.	2. Team members are actively engaged in the team's work.	2. Team members are actively engaged in the team's work.	2. Team members are actively engaged in the team's work.	2. Team members are actively engaged in the team's work.	2. Team members are actively engaged in the team's work.		

Template available at:  
<http://www.agileacademy.com.au/agile/sites/default/files/Quality%20Focused%20Project%20BVC.pdf>

#	Action	Done
1	Create cards for the quality attributes and associated criteria	
2	Agree on metrics and measures that will be used to track progress and make visible on a big visual chart (BVC)	
3	Complete Agile Quality Practices Assessment to identify quality practices targets	
4	Include quality attributes and associated criteria in definition of quality in test strategy	
5	Determine the automation strategy	
6	Revisit Quality definition in retrospective to confirm definition is still the same	

**INSTRUCTIONS**

*Agree any next steps you will do at as a team, either from issues raised in this workshop or from a maturity assessment as shown on the previous page.*

*Delete this instruction and replace the example above.*