Assess Your Agility

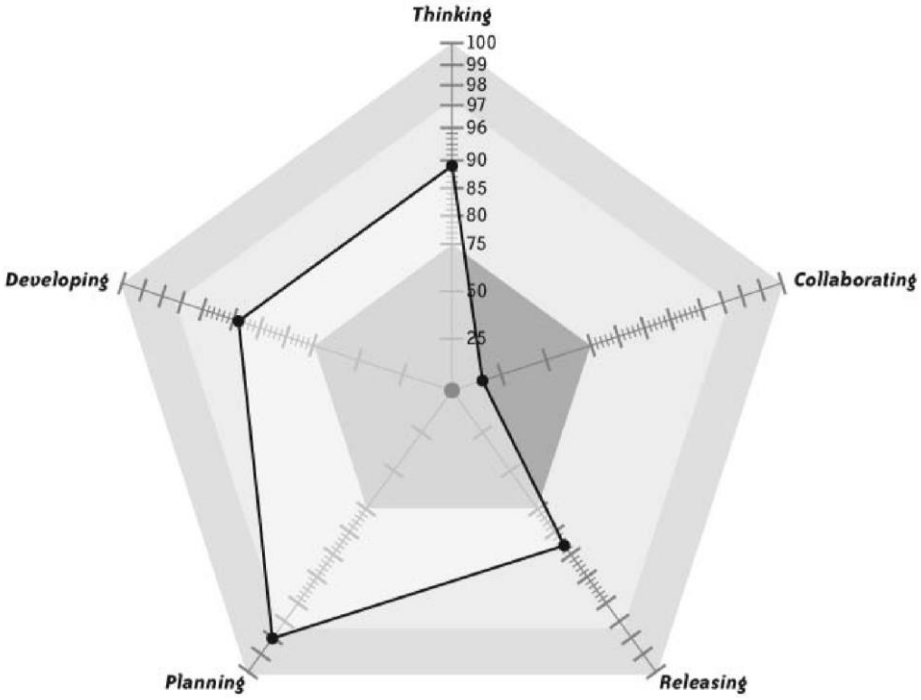
Document Description

This document is heavily based upon the suggested method for measuring team agility described in the Art of Agile Development. Each iteration it is advised that the team perform this test to provide a metric of agility and look for ways to improve.

Don't give partial credit for any question, and if you aren't sure of the answer, give yourself zero points. The goal should be to achieve the maximum score in each category. Any score less than the maximum indicates risk, and an opportunity for improvement.

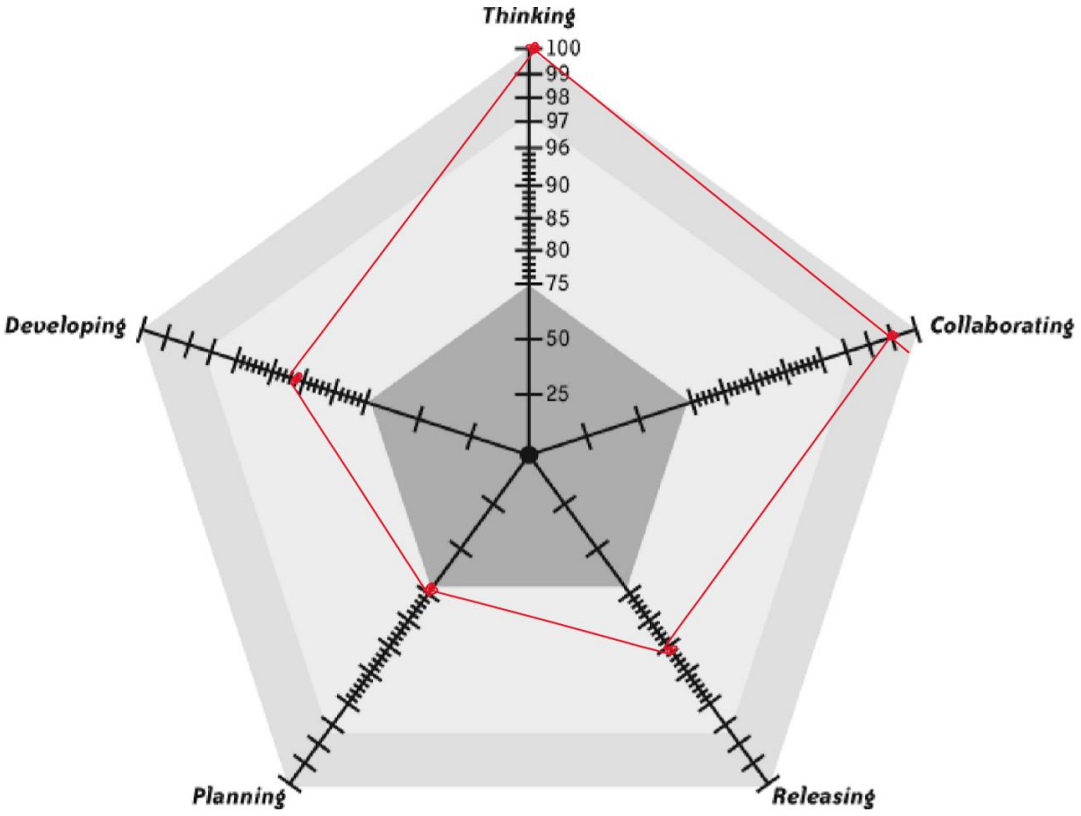
- 75 points or less: immediate improvement required (red)
- 75 to 96 points: improvement necessary (yellow)
- 97, 98, or 99: improvement possible (green)
- 100: no further improvement needed

Example Completed Test



Note: The point values for each answer comes from an algorithm that ensures correct risk assessment of the total score. This leads to some odd variations in scores. Don't read too much into the disparities between the values of individual questions.

Test Results Graph



Employee: James Gardner - Technical/QA Manager

Date of Completion: 130420

Signature:

Self-Assessment Quiz

Planning

Question	Yes	No	Methodology Under Test
Do nearly all team members understand what they are building, why they're building it, and what stakeholders consider success?	25	0	Vision;
Do all important stakeholders agree on what the team is building, why, and what the stakeholders jointly consider success?	25	0	Vision;
Does the team have a plan for achieving success?	4	0	Release Planning;
Does the team regularly seek out new information and use it to improve its plan for success?	2	0	Release Planning;
Does the team's plan incorporate the expertise of businesspeople as well as programmers, and do nearly all involved agree the plan is achievable?	3	0	Planning;
Are nearly all the line items in the team's plan customer-centric, results-oriented, and order-independent?	4	0	Stories;
Does the team compare its progress to the plan at predefined, timeboxed intervals, no longer than one month apart, and revise its plan accordingly?	4	0	Iteration;
Does the team make delivery commitments prior to each timeboxed interval, then nearly always deliver on those commitments?	4	0	'Done, Done'
After a line item in the plan is marked "complete," do team members later perform unexpected additional work, such as bug fixes or release polish, to finish it?	0	25	'Done, Done'
Does the team nearly always deliver on its release commitments?	3	0	Risk Management;
Total	**74**		

Collaborating

Question	Yes	No	Methodology Under Test
Do programmers ever make guesses rather than getting answers to questions?	0	75	The XP Team;
Are programmers usually able to start getting information (as opposed to sending a request and waiting for a response) as soon as they discover their need for it?	4	0	Sit Together;
Do team members generally communicate without confusion?	4	0	Sit Together; Common Language;
Do nearly all team members trust each other?	4	0	The XP Team; Sit Together;
Do team members generally know what other team members are working on?	1	0	Stand-Up Meetings;
Does the team demonstrate its progress to stakeholders at least once per month?	4	0	Iteration Demo; Reporting;
Does the team provide a working installation of its software for stakeholders to try at least once per month?	1	0	Iteration Demo;
Are all important stakeholders currently happy with the team's progress?	3	0	Iteration Demo; Reporting; Customer Involvement;
Do all important stakeholders currently trust the team's ability to deliver?	3	0	Trust; Reporting;
Total	**99**		

Developing

Question	Yes	No	Methodology Under Test
Are programmers nearly always confident that the code they've written recently does what they intended it to?	25	0	TDD;
Are all programmers comfortable making changes to the code?	25	0	TDD;
Do programmers have more than one debug session per week that exceeds 10 minutes?	0	3	TDD;
Do all programmers agree that the code is at least slightly better each week than it was the week before?	25	0	Refactoring;
Does the team deliver customer-valued stories every iteration?	3	0	Iterations;
Do unexpected design changes require difficult or costly changes to existing code?	0	3	Simple Design;
Do programmers use working code to give them information about technical problems?	1	0	Spikes;
Do any programmers optimize code without conducting performance tests first?	0	3	Performance Optimization;
Is there more than one bug per month in the business logic of completed stories?	0	3	Customer Tests;
Are any team members unsure about the quality of the software the team is producing?	0	1	Exploratory Testing;
Total	**85**		

Releasing

Question	Yes	No	Methodology Under Test
Can any programmer on the team currently build and test the software, and get an unambiguous success/fail result, using a single command?	25	0	Iterations;
Can any programmer on the team currently build a tested, deployable release using a single command?	5	0	Iterations;
Do all team members use version control for all project-related artefacts that aren't automatically generated?	25	0	Version Control;
Can any programmer build and test the software on any development workstation with nothing but a clean check-out from version control?	25	0	Version Control;
When a programmer gets the latest code, is he nearly always confident that it will build successfully and pass all its tests?	5	0	Continuous Integration;
Do nearly all programmers share a joint aesthetic for the code?	1	0	Coding Standards;
Do programmers usually improve the code when they see opportunities, regardless of who originally wrote it?	4	0	Collective Code Ownership;
Are fewer than five bugs per month discovered in the team's finished work?	1	0	No Bugs;
Total	**86**		

Ideals

Question	Yes	No	Methodology Under Test
Do programmers critique all production code with at least one other programmer?	5	0	Pair Programming;
Do all team members consistently, thoughtfully, and rigorously apply all the practices that the team has agreed to use?	75	0	Pair Programming; Root Cause Analysis; Retrospectives;
Are team members generally focused and engaged at work?	5	0	Energized Work
Are nearly all team members aware of their progress toward meeting team goals?	7	0	Informative Workspace;
Do any problems recur more than once per quarter?	0	5	Root Cause Analysis;
Does the team improve its process in some way at least once per month?	5	0	Reviews;
Total	**102**		