

The Agile Audit

This document can be used prior to a project starting (to determine if the Agile Process is suitable and how it might be applied) and during a project to review the practices in use. The objective is not to provide a “write/wrong” assessment but to stimulate discussion on how the project is operating and where it could be improved, perhaps as part of a retrospective/end of iteration. The check list draws influences from several Agile variants and ought to be suitable for any Agile environment.

1. Project Roles

Not all projects (due to small size or short duration) require all roles to be filled and in some cases, one person may undertake more than one role on a project.

<i>Role</i>	<i>Post Holder</i>	<i>Hours</i>
Business Sponsor		
Business Visionary		
Business Ambassador		
Business Advisor(s)		
Project Manager/Leader		
Business Analyst(s)		
Technical Architect/Lead		
Engineering Team Leader		
Software Engineer(s)		
Quality Assurance Engineer(s)		

2. Requirements & Technical Architecture

<i>Project Question</i>	<i>Project Statement</i>
Did the project have a Discovery/Prestudy Phase? What artefacts were produced, who reviewed them, what feedback was given to the team?	
What are the business benefits of the project?	
What is the expected Return on Investment?	
At what point does the cost of the project exceed	

<i>Project Question</i>	<i>Project Statement</i>
the business benefit?	
Have all of the requirements been prioritised?	
Who prioritises the features?	
Where can the requirements be found?	
How are changes (new features or unwanted features) identified?	
Beyond requirements, are any additional materials available to brief code and test engineers? (E.g. Activity Diagrams, Process Flow Diagrams, Screen Mock ups)	
Non functional requirements – have been identified, documented – where visible?	
How are technical requirements/stories handled? (Technical stories are often referred to as technical debt or sometimes they technical changes that have to be made but in themselves do not deliver any particular user requirement)	
Application Architecture? Defined up front or determined on the fly? Where is the architectural approach defined?	

2. Project Planning

<i>Project Question</i>	<i>Project Statement</i>
What Elements of the project are fixed? Features Time Resource/Cost	
Availability of Business Users for Iteration Planning?	
Does the project have an iteration and increment plan?	
Typical iteration and increment lengths?	
Do iterations and increments have their duration extended?	
Are the features to be developed in each iteration: a) fixed and not allowed to change b) roughly planned out and committed to	

<i>Project Question</i>	<i>Project Statement</i>
<p>shortly before the start of each iteration?</p> <p>c) randomly selected?</p>	
Exploration phase before the start of each new iteration or increment?	
Pre-planning prior to an Iteration Start?	
Is the Project using stories?	
Granularity of stories (too high, to granular)	
<p>Are visual controls used?</p> <p>(Cards on board/wall, burn down charts, software tool)</p>	
<p>Key Performance Indicators (KPIs) Used:</p> <p>Normal KPI's include completed (equals deployable) story points, Bug Count, Technical Debt, Code Quality (e.g. Copy/Paste Detector), Number of good/bad builds, Unit Test Coverage %, Regression Test Suite Size, number of successful tests, number of times a bug or story has pinged between Dev and QA)</p>	
How are KPIs gathered (e.g. readily available from a tool, manual calculation – work effort involved in collecting KPI's).	
Velocity (Include stories completed, retired, added, new estimates – are technical stories included? Are only completed stories counted towards velocity)	
How are technical stories handled?	
<p>Does the BA or Business User record additional information on the story card and story management system? (e.g. programmers/QA engineers notes, key points made during conversations).</p> <p>Lesson learned from projects; story notes are good.</p>	
Does the BA (or story manager) work on requirements for an iteration ahead of the iteration?	
<p>Prioritisation of Technical Debt?</p> <p>(Is technical debt recorded, if so as a story or bug?)</p>	
<p>How are Stories Estimated?</p> <p>Spikes, prior work on other projects or this</p>	

<i>Project Question</i>	<i>Project Statement</i>
project? Comparison between stories of equal complexity.	
How accurate are estimates?	
Are the minimal features required or a usable delivery known? How arrived at/validated? Critical path through system (wide coverage shallow feature set or narrow coverage with many features?)	
Is the Backlog of work fully estimated?	
When users bring new features/changes to the project: <ul style="list-style-type: none"> a) Rejected out of hand? b) Are they discussed and estimated? c) Are the users required to trade an existing features for the new feature so that the overall backlog does not increase in size? (If not how are Iteration and Increment Timeboxes being maintained?) 	

3. Communication

<i>Project Question</i>	<i>Project Statement</i>
Are daily stand ups held? (Length)	
End of Iteration/Retrospective?	
Does Retrospective add any value? (If no, then something is very wrong – change the format urgently!)	
Does the Retrospective review KPI's? Are the meetings noted and are actions from last meetings reviewed for follow up?	
Use of Wiki and Project Forum to share data and other information?	
Use of notice boards?	
Huddles at start of a story (BA/Customer, Code and Test Engineers)	

<i>Project Question</i>	<i>Project Statement</i>
Does the QA Engineer and BA review stories during coding and again prior to hand over for formal QA?	
Reporting to business and other interested parties (e.g. Project Office). Format and value?	
JAD Workshops with users?	
Email? (does everybody have it, do they need it?)	
Ability to access reference materials on business and technical subjects.	
Communication to related teams (e.g. production) and other programme teams. How and frequency.	
Do the team ever use Quality Circles to review progress?	
Is the team located in the same workspace ? (if in the same building)	
If the team is located in different locations/timezones, how is interactive communication achieved?	
How often (and when) do business users review deliverables – e.g. a “show and tell” at the end of each iteration?	
Level of Engagement from Business Users during Iterations?	
Are release notes produced and circulated to interested parties?	
Communication with Operations Team in readiness for Deployment and/or provisioning of new hardware for the project?	

3. Code & Build

<i>Project Question</i>	<i>Project Statement</i>
Is a source management system used? (CVS, Subversion)	
Automated Build System? (e.g. Cruise Control, Hudson/Jenkins)?	
Who owns and maintains the build system?	
Does the build system produce a ready to deploy application?	

<i>Project Question</i>	<i>Project Statement</i>
(Important so allow anybody to take and use any build; particularly for projects where there are limited options to put iteration deliverables into production)	
Are builds version numbered? Are build numbers linked back to the requirements management system so that features per build can be easily identified?	
Can a build be recreated several days after being delivered?	
How often does the build run? (How long does it take?)	
How often do engineers check in?	
Prior to check in, Do engineers have to <ul style="list-style-type: none"> a) Ensure their code is synchronised with other team members? b) Ensure that is passed all units tests & compiles locally? 	
Is pair programming used (intelligent or all the time). If not, how is knowledge of the code base shared to reduce dependencies on one person?	
Are any coding standards/conventions used? (If so how monitored and where are they set out (e.g. Wiki)).	
How are areas for refactoring identified?	
How is refactoring managed ? (e.g. adhoc as programmers find it or by a story card included as a technical debt story?)	
Is the application design continually reviewed?	

4. Test

<i>Project Question</i>	<i>Project Statement</i>
Is the QA Engineer involved in the planning game?	
Does the QA Engineer discuss/review unit tests with the coding Engineers? (Part of code quality/assessing value of unit tests, communication on how features have been implemented) If so, how recorded (ISO/CMMI	

<i>Project Question</i>	<i>Project Statement</i>
certification)	
What test scripts (other than unit tests) are produced?	
Aside from unit tests, who performs QA work on the project – do Developers assist with QA?	
Are regression tests included the automated build process?	
How often are they run outside the build?	
Does the BA or Customer Representative review work before being handed over to QA for test?	
Does the BA and/or Customer Representative help devise Regression Tests	
How are defects recorded? (On card and in Bug Tracking System)	
Do Engineers take bugs in preference with new stories?	
Are manual QA Scripts maintained	
Are QA Artefacts (Test Data, scripts, notes etc) stored in a repository?	
Who conducts UAT?	
Are UAT written by the business users?	
Are there any performance targets for the projects deliverables?	
How and by whom are deliverables measured for performance?	
Are security requirements identified in test scripts?	
How are test tested (e.g. Penetration Test for a web application)	
Does the project have access to a separate test area that mimics production?	
Does the project have tested and ready to deployable software at the end of each Iteration that could, in theory, be taken to Production if the Business so desired?	
Recognising that the project may produce several	

<i>Project Question</i>	<i>Project Statement</i>
iterations of work before a formal release to production, how are interim end of Iteration deliverables tested to ensure continued ability to deploy to production?	

5. Release & Project Documentation

<i>Project Question</i>	<i>Project Statement</i>
Is the target release environment understood – e.g. O/S, App Server – patch versions etc?	
Is the target release environment designed to be High Available (HA) and/or Disaster Recovery (DR)? How does this affect the design and release documentation?	
What handover documentation will the operations team require?	
Are release and install instructions produced for each release? Who is responsible for producing these?	
Are any user instructions/help materials produced, if so by whom and when (during the iteration?)	
Are releases packaged into an installer for ease of deployment by users?	
Does the install package/release include any tools for ease of upgrade from earlier versions?	
What documents are produced by the project? Why are these produced (company standards, somebody will use them – a project should only product material that will be of value to somebody). End of Projects Docs: Functional Overview, Business Processes used in the software, ERD, Use Cases, Architecture Diagrams, Configuration Diagrams, plus all release notes and end user docs.	
Are the contents of the Wiki/Intranet page for the project considered project documentation?	
After the project has been completed, who will maintain the deliverables and what documentation/hand over process will they require?	