ASSIGNMENT:

Paper: Software Development Lifecycle Model and Testing

Length: 2.5 - 3 pages double spaced

Topic: Describe a software lifecycle model and how testing is a part of that model

Pick a Model: V-Model - Waterfall - Spiral - Agile - Iterative – (Incremental) – (Scrum)

List sources at the end of the paper.

Questions you may want to answer:

What are the pros and cons of that model?

How is testing organized/used/implemented in that model?

What are the challenges and benefits of testing in this model?

PAPER:


This paper begins by explaining process modeling principles that should be employed for a modeling process to be “agile”. The second part of this paper points out problems with real world application of agile and how Disciplined Agile Delivery “DAD” can improve process and product. Finally, the problem of how to organize and move forward with projects of scale is addressed. The solution posed is for the team to use software such as IBM Rational Team Concert, discussed briefly. This paper includes a space consuming illustration and an “Aside”. Therefore the length runs slightly more than three pages.

Agile process, applied according to agile principles, has brought about product quality improvements, increased efficiency and timely delivery of small projects. This has caused people involved in projects of scale to work on adapting agile principles to apply to larger scale projects. Among those principles are these.

1. People are the primary determinant of success for the delivery of IT projects
2. Moving to an agile delivery process is the first step in scaling agile strategies
3. Disciplined Agile Delivery (DAD) is an enterprise aware, hybrid software process framework.
4. Agile strategies should be applied throughout the entire delivery life cycle.
5. Agile teams are easier to govern than traditional teams. [Reference this site, page 2] https://books.google.com/books?hl=en&lr=&id=CwvBEKsCY2gC&oi=fnd&pg=PR11&q=Agile%20DAD%20for%20large%20scale%20projects&f=false

This process model emphasizes delivery of working, high business value software quickly by using teams rather than isolated individuals or discrete groups such as a developer group, a testing group, management.[Reference]http://www.testingexcellence.com/category/agile-testing/

At development life cycle inception, the vision of the product owner, usually a manager, is documented. An initial team is assembled to include the owner, technical people, users and
representative members of other stakeholders. User stories around the product owner vision are collected, often from the product owner or an analyst. [Aside comment appears above References] The qualities of a good user story is that it be independent, negotiable, valuable, estimable, small and testable. These stories, chunks of user needs, are placed in a list called a backlog. They are “groomed” and prioritized by the product owner, hopefully with user input when advisable. The grooming process converts user stories from particular needs into a form that can be coded and tested. The resulting version of the user story must contain user acceptance criteria. They will be used later for QA acceptance testing and designation of an iteration, “Done”. User stories about how a product will be used, groomed and prioritized by the product owner, constitute the product backlog list.

At this point a series of "plan-develop-test-implement-review" iterations begins, commencing with the top (highest priority) item, possibly including related items, on the list. Agile terminology for iteration is “sprint”. Prior to each sprint there is a short team meeting for self-organization around various development and QA testing for that sprint. But before self-organization occurs, there is a short review of the work done during the prior sprint. This ensures that at the end of that sprint a product will be ready and that item on the backlog can be labelled “Done”. A sprint is not complete until the backlog item(s) being addressed can be so labelled.

It should be added that the combining of more than one backlog item in a single sprint occurs only when the logic of each item closely relates to the other(s). That could be restated as "is on the same developmental level as the others and has related functionality". At the end of each sprint a product (not the final product) will have been created that is ready for acceptance testing against acceptance criteria previously included in the user story on the backlog.

As stated above, before the team can start a new sprint there must be a review, a retrospective, of what was accomplished during that sprint just completed. This review is conducted at the start of a short, early morning meeting, hopefully occurring every day. The review may well call for backlog changes, revised requirements or change requests to be sent to the product owner or a governance entity. After the review has determined any change needs, self-organization for doing the work of the next sprint is commenced.

Documentation is minimal, usually consisting of an operations manual and system overview documents. In the agile development scenario the role of tester is to find the potential for defects in the software, not to test exhaustively for bugs of all sorts. Under agile conditions, testing strategy is to include a mission statement, a build that conforms to wider business goals and testing best practices. QA testing is defined as all the activities during product development performed to ensure correct quality and intended to satisfy customer acceptance criteria. QA is the responsibility of everyone on the team, not just those team members who are performing most of the QA testing. An illustration of the various types of testing that occur during software development appears below. [Reference] http://www.testingexcellence.com/whole-team-approach-agile-testing/
This site contains a good but also lengthy list of QA tester guidelines. [Reference] http://www.testingexcellence.com/software-testing-practical-tips-for-software-testers/ .

Although more companies report that they use agile principles in the process of creating software than any other SDLC, according to the site below, the degree of adherence to agile modeling criteria is far from universal among these companies. Moreover, data show that project owners and project managers who report that they use any other iterative processes in software development achieve about the same product success rate as those who claim to adhere to an agile process. This is where upgrading agile software development process to DAD could provide considerably better success rates for agile over iterative and other methodologies. According to this agile modeling site, these five criteria are pretty good determinants for measuring degree of adherence to DAD (improved) agile modeling principles. [Reference] http://www.agilemodeling.com/essays/agileCriteria.htm

1. Does the team regularly produce value (software) for their stakeholders and are iterative milestones met? [See Figure 1. Agile criterion: Value – A disciplined team will produce a consumable solution each iteration or sprint.]
2. Does the team validate their work (test each small bit or condition) to the best of their ability? [See Figure 2. Agile criterion: Validation - Observe Test Driven Development “TDD” by writing a single test before writing code to fulfill just that test."
3. Are stakeholders actively involved? [See Figure 3. Agile criterion: Stakeholder Involvement – They should be providing information every day.]

4. Is the team self-organizing or is task allocation done from above? [See Figure 4. Agile criterion: Self Organization and Appropriate Governance - DAD agile teams work within the context of an effective governance framework which guides and monitors team efforts. This includes working toward a common infrastructure typically driven by agile enterprise architecture and common programming guidelines.]

5. Does the team strive to improve their process? [See Figure 5. Agile criterion: Continuous Improvement – The team should hold a retrospective at the end of each iteration to identify potential ways to improve their process. They should act on one or more of these issues in the next iteration.]

Shortening feedback cycles, the time between outputting code and getting feedback, is another DAD technique that could lower risk, cut time and improve process. The site below offers these suggestions. Program in pairs, developer and tester. Receive daily in-person input from stakeholders, using an online meeting app for those who may be located at some distance. Build; regression test; run through code analysis on a daily basis. Continually deploy from the development environment into the product integration environment, a more complex environment. To prevent time wasting activities from creeping in, make iterations/sprints as short as possible with the caveat that large projects could require longer sprints.

https://books.google.com/books?hl=en&lr=&id=CwvBEKsCY2gC&oi=fnd&pg=PR11&dq=Agile+DAD+for+large+scale+projects&ots=lFY63gT-HF&sig=gqw9K0315JqR1yWrlMKLuj26mfY#v=onepage&q=Agile%20DAD%20for%20large%20scale%20projects&f=false

Finally, there is the problem of scaling agile methodology to large scale software development. IBM appears to be the exemplar company here with advanced lifecycle management software solutions.

**IBM application lifecycle management** solutions bring business, development and quality assurance teams together to continuously adjust business goals based on customer feedback. They help application development teams stay closely aligned throughout the lifecycle to deliver projects faster and with more customer value. These solutions provide a scalable, integrated and open application development platform that supports lead and agile approaches, such ... Scaled Agile Framework® (SAFe®). Diverse teams, including developers committed to keeping their current tools and those working with integrated cloud platforms such as IBM Bluemix® can be accommodated. [Reference] http://www-03.ibm.com/software/products/en/rtc

IBM® Rational Team Concert™ 6.0 includes the integrated Scaled Agile Framework® (SAFe®) template. It’s an agile application life cycle management solution that keeps development teams connected, manages code, runs standups (pre-iteration and post-iteration meetings), assists in sprint planning and tracks work. It accommodates planning and reporting on a common platform. [Reference] https://jazz.net/products/rational-team-concert/?cm_mc_uid=32274044905114748272258&cm_mc_sid_5020000=1474829460

A drawback to the agile model has been shown to be lack of clarity and definition as to how to rigorously implement Agile principles. With the development of DAD, that is no longer the case. DAD offers a process upgrade check list and expert advice for successful implementation of agile principles. Another drawback to the agile model was shown to be the complexity and sheer number of human beings interacting in the development of large products. IBM offers
advanced software to augment human efforts in just about every facet of large scale production using agile principles.

Referring to the Agile Testing Quadrants figure above, a QA Tester would probably spend most of her time working the “Critique Product” side of the figure, especially doing “ility” tests, in coordination with the developer, with guidance from the Team Leader. This writer does not see any challenges to testing in an agile setting. In fact, testing within a team, ideally with a developer partner, seems to be the best of all possible settings.

Therefore in closing one could conclude that agile principles and methodology has been successful, especially as refined by incorporating DAD principles.

- Agile uses the benchmarks inherent to an iterative approach to save time by cutting down on unnecessary, duplicative work and in waiting for validation feedback.
- Agile principles encourage cross-training among members of a team thus increasing team members’ versatility and an overall more flexible team that is better prepared to adapt to changes in requirements.
- Agile increases a positive spirit of collaboration that may positively affect overall company morale.
- Agile decreases project complexity by breaking it up into understandable, manageable chunks and this decreases risk.
- Agile produces product faster as each iteration results in a deliverable and, hopefully successful integration. Happier management and happier customer results.
- Agile training for company staff is readily available due to the popularity of this SDLC.
- Agile DAD incorporates governance more explicitly so self-organization isn’t totally without logic; change requests have someplace to go; there is a “go-to” for interpersonal issues; events that could distract the team can be deflected.

Aside: Based on the workplace experience of this writer, this accepted source for user stories would not be optimum in some situations. It is critical to include stories from people who use or will use and rely on the system being developer as part of their every-day job. It’s often people on the bottom bureaucratic rungs who are most intimately familiar with the current system or have the best grasp of their needs for a new or revised product. Without their input, critical user stories often are omitted. This is especially important when the users are mostly in one organizational unit but the developers, QA specialists, analysts are in another unit or working on contract. Added complexity comes when the two units have historically operated in each other’s cross hairs.

REFERENCES in order of appearance in this paper.

https://books.google.com/books?hl=en&lr=&id=CwvBElkCy2gC&oi=fnd&pg=PR11&dq=Agile+DAD+for+large+scale+projects&ots=IFY63qT-
HF&sig=ggw9K0315JqR1yWrlMKLuj26mfY#v=onepage&q=Agile%20DAD%20for%20large%20scale%20projects&f=false

http://www.testingexcellence.com/agile-test-strategy-example-template/
http://www.testingexcellence.com/software-testing-practical-tips-for-software-testers/
http://www.agilemodeling.com/essays/agileCriteria.htm
https://jazz.net/products/rational-team-concert/?cm_mc_uid=32274044905114748272258&cm_mc_sid_50200000=1474829460