Chapter 5. **Introduction To The Process**

How is this process different from other agile processes?

In this chapter, I provide an overview of the process, its relationship to Scrum and how it expands on the existing Scrum activities and artifacts.

The Quality with Agile through Pictures process is the result of real environments that I have worked in as a business analyst. Not all of those environments adopted all of the activities described in the process; however, every development effort used a Scrum like process (with user stories and fixed sprints).

- **Because every project has at least 1 business analyst (me), the Elicit Business Needs and Maintain Requirements activities have always been adopted, as a minimum.**

The environment in which these projects are conducted, include one or more of the following traits:

The customer is remote – when developing a commercial product, there is always an offsite customer. In this case, the product owner (or similar role) acts as a proxy for the customer.

The development team is remote - many companies do not have a co-located IT department. Sometimes software development is subcontracted to an external company. In both cases developers work at their location which is not the same as that of the business.

Quality assurance is a separate department – the quality assurance department is a separate entity with management that is detached from the development. The quality assurance department is a shared resource across the enterprise, and it may be remotely located.

UX experts design a common user interface – the user interface designers may be within a UX department that is not only responsible for software user interfaces, but also supports marketing and company branding. All products share the product user interface designs.

Software deployment - deployment often requires integration with hardware and software components that are already installed at the customer location. Software cannot be deployed to a customer site until its quality has been approved. The deployment manager is a customer facing role and not considered part of the software development team.

Documentation - a software release includes release notes and user manuals written for both internal and external stakeholders. The documentation is written for a customer release and not for each incremental build.

The result is that additional activities and roles are added to the Scrum development process to accommodate each of these situations. When an activity is adopted to overcome one of these traits, a responsible role is associated with that activity, and the activity deliverables are identified.

5.1 **Scrum Process Framework**

Figure 6 shows a diagrammatic representation of the Scrum framework. This represents a typical Scrum process (that I borrowed from the Scrum framework poster at https://www.scrum.org/resources/scrum-framework-poster).

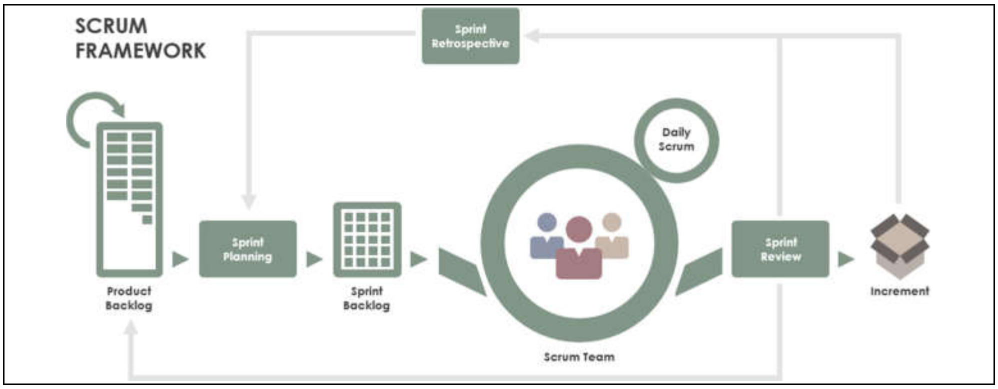

Figure 6 - Scrum Process Framework Diagram

The Scrum process framework shows a product backlog containing user stories. These user stories are taken into a sprint planning meeting. The sprint planning meeting populates a sprint backlog with user stories from the product backlog. The (Scrum) development team uses the user stories in the sprint backlog to create an incremental build (increment).

A daily Scrum meeting is held by the development team. The incremental build is reviewed at the end of the sprint. Finally, a sprint retrospective is held in order to capture lessons learned from the sprint.

For the purpose of using a consistent notation, I have drawn the same process using the process flow diagram notation described in Appendix A - . Figure 7 uses activities (use cases) to show the work performed in a Scrum and it uses data flows to show the information passing between those activities.

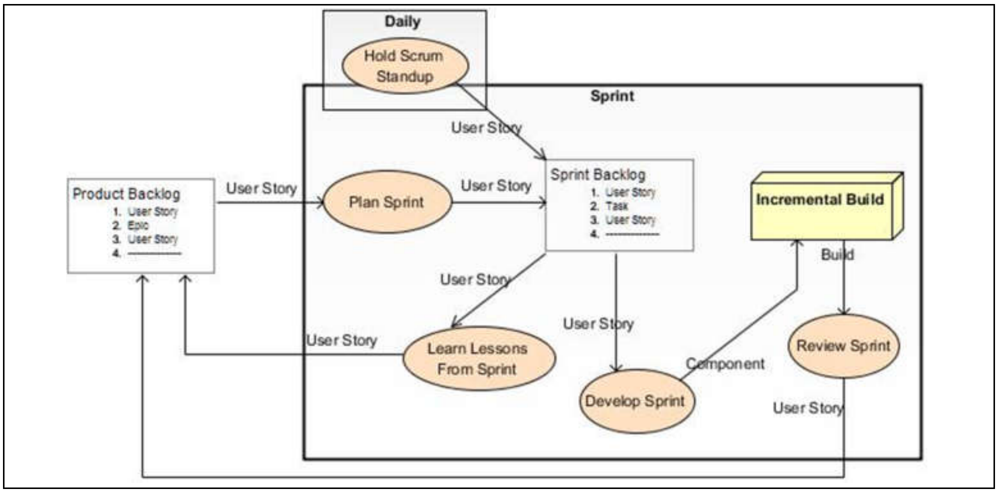

Figure 7 - Scrum Framework Process Flow Diagram

Scrum defines activities that occur within a sprint. A sprint has a set of goals that a development team attempt to achieve over a fixed period of time. The product backlog is the input to a sprint. An incremental build is the output from a sprint (which satisfies the sprint objectives). User stories represent the information that flows between the sprint activities.

In order to be able estimate work and accurately measure the effectiveness and any improvements in development, every sprint should provide the same number of working hours. (In reality, sprints are planned to occur on a weekly, bi-weekly or monthly basis, without taking into account the actual number of working hours in the sprint.) No matter how your sprint is planned, it should last no more than 1 month in real time. After 1 month, the original goals of the sprint become blurry, and requirements are changing.

Keeping the sprint short allows the team to anticipate when people are not going to be available and plan for less development hours as necessary.

A longer sprint allows for more upfront planning and less Scrum meetings, but more chance that your sprint will not go according to plan.

♦ **As an aside, I researched recommendations for sprint planning over the long holiday periods.**
Recommendations are:
- extend the sprint by the number of missing days.
- keep the same real-time sprint length and adjust the planned work and velocity measurements to accommodate the missing hours.

My suggestion:
Extend the sprint by a full week and give the development team additional days off from development to make the holiday into 5 working days.
You are effectively removing that week from your calendar. This gives the developers a break from coding and allows them to catch up on training, or other activities that they might like to do if given a break from developing.
Scrum practitioners appear highly discourage changing the length of a sprint. By

removing a week from your Scrum calendar you are keeping the sprint the same length and, but introducing a gap into the sprint.
If introducing a gap into a sprint is unacceptable, then you could plan you sprints so that they do not fall on a break in the calendar.

5.1.1 Activities

The activities in Scrum are called sprint ceremonies. These are renamed as follows to reflect what the activity does:

- Plan Sprint – This activity is generally in the form of a meeting prior to each sprint where the sprint goals are defined and the sprint backlog is populated with user stories.
- Develop Sprint – This activity represents the work done by the development team in order to meet the goals of the sprint.
- Review Sprint – This activity is performed at the end of the sprint. The results of the sprint are reviewed in order to determine if the goals were met.
- Perform Sprint Retrospective – This is a meeting that is held after the sprint is complete. (Often it occurs during the next sprint.) It allows all people involved with the sprint to address issues with the last sprint and discuss the best practices that should be carried over to future sprints.
- Conduct Daily Standup – This is meeting held every day, where the previous day’s progress is discussed and issues are addressed. This meeting occurs every day whether there is an active sprint or not, and all interested stakeholders are invited to participate (which is why this activity is shown both inside and outside the sprint).

5.1.2 Artifacts

Scrum identifies 4 artifacts (Incremental Build, User Story, Product Backlog, Sprint Backlog). The Scrum Framework Process Flow diagram shows 3 additional artifacts (Epic, Task and Component). The complete set Scrum of artifacts are:

- Component - Software that is delivered as the result of a user story. Many software components are delivered in a sprint.
- Epic – An epic is a user story that is too large to be completed in a single sprint. Epics are only found in the product backlog.
- Incremental Build – The product that is updated with components from a sprint.
- Product Backlog – This is a prioritized list of user stories (and epics). There is 1 product backlog per development team.
- Sprint Backlog –An ordered list of user stories (and tasks) to be completed in a single sprint. There is 1 sprint backlog per sprint.
- Task – The breakdown of a use story into work that may be performed in a single day.

- User Story – A requirement to be developed (and tested) within a single sprint.

User stories in the product backlog are taken into the next sprint backlog during plan sprint activity. User stories in the sprint backlog are developed into software components. Software components are combined into an incremental product build at the end of the sprint.

User stories in the sprint backlog are updated during the daily standup.

The sprint review may generate new user stories for the product backlog.

The sprint retrospective meeting may generate new user stories for the product backlog.

5.2 **A Common Scrum Process**

Agile recommends collocating development teams with customers and all supporting roles. In this manner, a developer is able to communicate with the right people to obtain the information they need at any time.

Even when the project has a collocated development team that includes documentation, user interface design, testing and solution architects, the Scrum process framework diagram is missing some activities.

In reality, the customer is almost never collocated with the development team. In this situation, the product owner acts as a proxy for the customer. Every Scrum project includes a product owner. The product owner is responsible for the accuracy and priority of user stories. The business analyst works closely with the product owner in order to assist with supplying accurate user stories to developers. Even though the product owner and business analyst are collocated with the Scrum team, their work is performed outside of the sprint cycle. This is why the activities that are the responsibility of these roles are shown outside of the sprint in Figure 8.

Backlog grooming is the responsibility of the product owner and it occurs whenever convenient. This activity is performed independent of the sprint cycle.

For every project where a business analyst role is involved, there will be an Elicit Business Needs activity. Eliciting business needs occurs at the convenience of the customer and is independent of the sprint cycle.

In addition to these activities, every Scrum project that I have been involved with deploys incremental builds on an independent release cycle. Several sprints may occur between releases of the product.

Figure 8 shows a Scrum process that includes these activities.

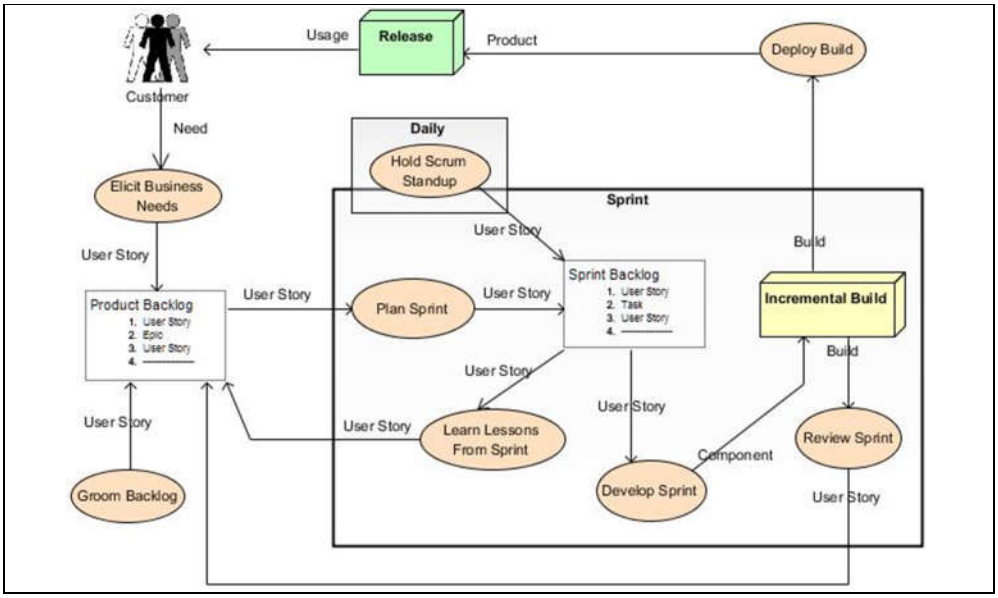

Figure 8 – Extended Scrum Process Framework

This extended Scrum process includes the Scrum framework described in Figure 6 and provides an additional 3 activities and 3 artifacts.

5.2.1 Activities

These additional activities are:

- Elicit Business Needs – The product owner captures business needs as user stories (or epics) in a product backlog. The business analyst adds requirements (in user stories) to the product backlog. The business analyst elicits the business needs from the stakeholders and writes these as user stories that capture the intent of the epics as they relate to a solution.
- Groom Backlog – As more user stories are added to the product backlog existing user stories need to be re-prioritized. User stories that have been in the product backlog for several sprints are re-visited, in order to determine if they are still accurate or even relevant.
- Deploy Build – This activity introduces a deployment manager who is responsible for updating customer systems with new releases of the product. (I have never experienced a project where the customer software is changed when the Scrum team is ready to make that change.) An incremental build is validated and approved prior to being deployed. The deployment manager works closely with the customer to deploy software builds at the appropriate time.

♦ **In my experience, deployment occurs when it is convenient for the customer, not the development team.**

5.2.2 Artifacts

The additional artifacts that are included in a typical Scrum process are:

- Need – This is a request from the customer.
- Product – A product is an incremental software build that has been approved as ready for deployment.
- Release – A release is a product and its support that is deployed at the customer environment.

The business need comes from the customer. This is converted into an epic and user stories in the product backlog. These epics and their user stories are prioritized according to order in which they will be considered for development. When an incremental build is available, it may be deployed to a customer as a product release.

The artifacts are created independently of the sprint cycle.

5.3 **Quality With Agile Process**

In addition to a typical Scum process, many activities might be added as a result of independent teams that contribute to the product. These independent teams include:

Quality assurance – This is often an independent organization separated from the development organization.

Solution architecture – This is often a separate team of architecture specialists who design the architecture of systems on which the software is built. This may be more common when several related software builds are integrated into a single solution.

Technical writers – These are a team of documentation specialists who produce user instructions for a product.

User interface design – This is an independent organization that is responsible for a consistent look and feel to customer facing products throughout the organization.

The Quality with Agile through Pictures process assumes that each of these roles is separately located. If your organization collocates any of these teams then the activity for which they are responsible may be removed as a separate entity and incorporated into the sprint as part of the Develop Sprint activity.

However, (as mentioned in section Chapter 4 on Roles) organizations often separate these roles on purpose. One reason for this separation is to encourage a higher quality of product.

Figure 9 is a diagrammatic representation of a process that assumes that these roles are separated or distinct organizations and that they operate outside of the sprint cycle.

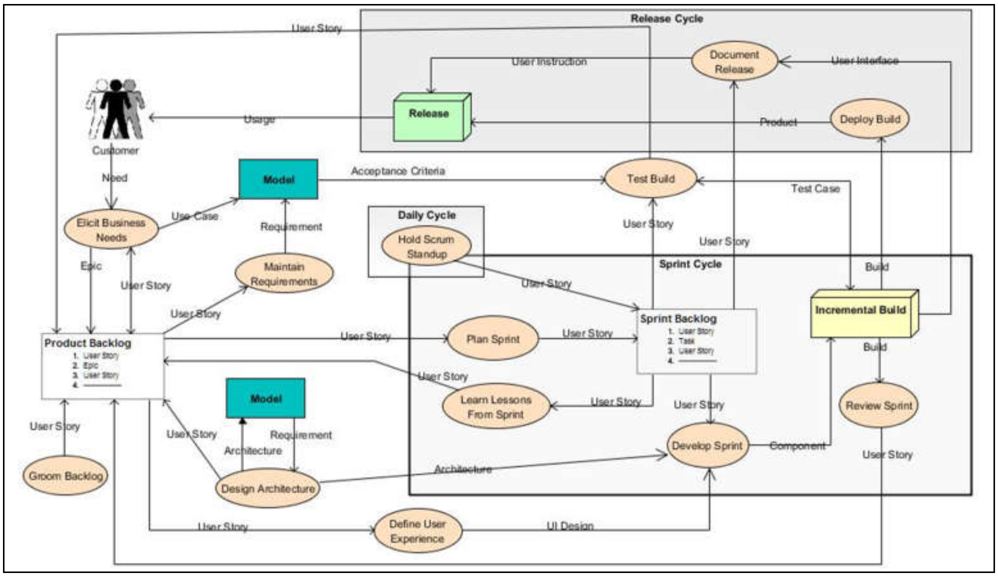

Figure 9 – Agile Though Pictures Process Flow Diagram

The Quality with Agile through Pictures process introduces 5 activities and 7 artifacts to the Scrum framework. It also introduces a release cycle that is independent of the sprint cycle.

5.3.1 Activities

There is no specific sequence specified in the process. Each activity may start when its appropriate inputs are available. The driving force behind the whole process is the sprint. Sprints are planned in advance according to a consistent and repetitive schedule. Activities providing inputs to the sprint need to supply enough information prior to the start of the sprint for it to be successful. Activities using the outputs from a sprint are dependent on the next scheduled release.

- ♦ **Releases are often scheduled on a consistent repeating period. Once a quarter is typical.**

These additional activities are:

- Design Architecture – The system architecture is designed by the solution architect. The solution architect is an independent role that supplies architecture information to the development team as the requirements dictate or as new technology becomes available.

- Define User Experience – User interface standards and designs are maintained by the UI designer. The UI designer is an independent role that provides a consistent look and feel to all company products.
- Document Release – User instructions for a release are compiled by the writer. Since documentation is supplied with a release, this activity is dependent on the release cycle.
- Maintain Requirements – A model of the to-be system is maintained by the business analyst. This is an ongoing activity that the business analyst performs as necessary or as time permits.
- Test Build – Product validation testing is performed by quality assurance. Quality assurance is an independent role whose primary objective is to validate that software incremental builds include sufficient quality to be released to the customer. It is assumed that this is a continuous activity that is not dependent on the sprint cycle.

5.3.2 Artifacts

The artifacts produced as a result of these quality activities are:

- Acceptance Criteria – These are created during analysis of the use cases. These are captured in the model and may be documented in the user stories.
- Architecture – The system architecture is designed by the solution architect and maintained in the model of the system.
- Model – The model is a repository for the artifacts that are created by the business analyst as a result of analyzing the business needs. It includes use cases, acceptance criteria and data, as well as supporting diagrams. It also includes a model the system architecture.
- Requirement – A requirement is any component of the model with which the system must be compliant. This includes use cases, acceptance criteria and data.
- Test Case – Test cases are created by quality assurance and used to validate an incremental build.
- Use Case – The business analyst captures business needs with use cases. They are maintained in the model.
- User Instruction – User instructions are a form of documentation that is produced by a writer and used by stakeholders to understand a product release.
- UI Design – This is a description of screens that are displayed to users who operate the system. Maintained by the UI designer they are often in the form of a detailed mockup diagram.

The flow of artifacts through the process follows the following sequence.

- **Artifacts may be returned to a previous activity at any stage in the process.**

The product owner populates the product backlog with user stories in the form of epics. (As the business analyst, I may create the epics in the repository from discussions with the product owner.) The business analyst elicits needs from the business stakeholders, based on those epics. The business analyst analyzes these needs on order to add detail to the epics and create user stories. The business analyst captures business actors, use cases and business activities in a model of the system. This model is maintained by the business analyst. It also includes data, non-functional requirements and the system architecture.

Acceptance criteria are derived from the use cases and used by testers to generate test cases. These test cases are used to validate an incremental build. Acceptance criteria are also included in the user stories to provide details to the developers.

The solution architect is responsible for the system architecture. This system architecture is captured in the model and may be maintained by the business analyst.

The user interface designs are supplied by the UI designer. User interface specifications are normally managed by a UI design tool. (UI design tools are not discussed in this book.)

The development team uses the system architecture and user interface design in conjunction with the user stories to produce an incremental build.

Testers validate the incremental build against their test cases and the result may be deployed as a release. User stories may be created in the product backlog as a result of defects found during testing.

A release includes user instructions in the form of documentation. These instructions are produced by a writer, from the user stories and by examining the product release.

5.4 **Defects**

In the Quality with Agile through Pictures process defects are captured as user stories.

Defects capture undesirable system behavior. They may be discovered in an incremental build or in a product release. Defects may be the result of software that does not meet the specified requirements (commonly known as bugs) or the result of requirements that do not accurately capture the business need.

Defects are generally discovered by quality assurance as the result of validating an incremental build. Once a feature has passed quality assurance testing, then it is assumed free from defects. Any changes to a product release are considered enhancements to the product, even if they are the result of undesirable behavior. Therefore, these enhancements are captured by user stories.

Although defects are of a different format to a typical user story, their Scrum life cycle is identical. They are entered into the product backlog, prioritized and pulled into a subsequent sprint.

Defect or enhancement type user stories may be discovered during any of the following activities:

- Deploy Build – A problem occurs with the system during deployment.
- Document Release – The writer may find inconsistencies between the product release and its user stories.
- Review Sprint – Defects may be found by stakeholders while reviewing the output of a sprint.
- Test Build – An incremental build fails during quality assurance testing.

In summary, defects are considered enhancements to the product and follow the same process as a user story that is created from a requirement.

5.5 **Kanban**

A few words about using Kanban to manage user story development.

♦ **I have little experience with Kanban, so this will be short.**

Instead of user stories being pulled into development via an activity such as sprint planning, a Kanban process continuously pulls user stories into the development team backlog. Work is limited by a work in progress (WIP) maximum. This WIP limit restricts the number of user stories that may be in a development at the same time.

Because there is no sprint, it is assumed that there is no sprint planning, no sprint review and no lessons learned activities in a Kanban development process. Since these activities occur within a sprint cycle the Quality with Agile through Pictures process remains the same outside of the sprint region.

The same work products are delivered by the same activities. The product backlog is the same, as are its epics and user stories.

A model of the product is maintained by the business analyst. Maintenance of the model is the same as described previously.

The architecture and UI design are still provided to developers as needed.

Product releases occur according to a release cycle and documentation is part of the release.

The only change is that the sprint backlog is replaced by a WIP backlog.

- ♦ **WIP limit appears to be based on the number of user stories and not the workload. I assume the reason is to keep each person in the development team working on exactly 1 user story at any time. This is to minimize multi-tasking.**

The Quality with Agile through Pictures process might look similar to that shown in Figure 10.

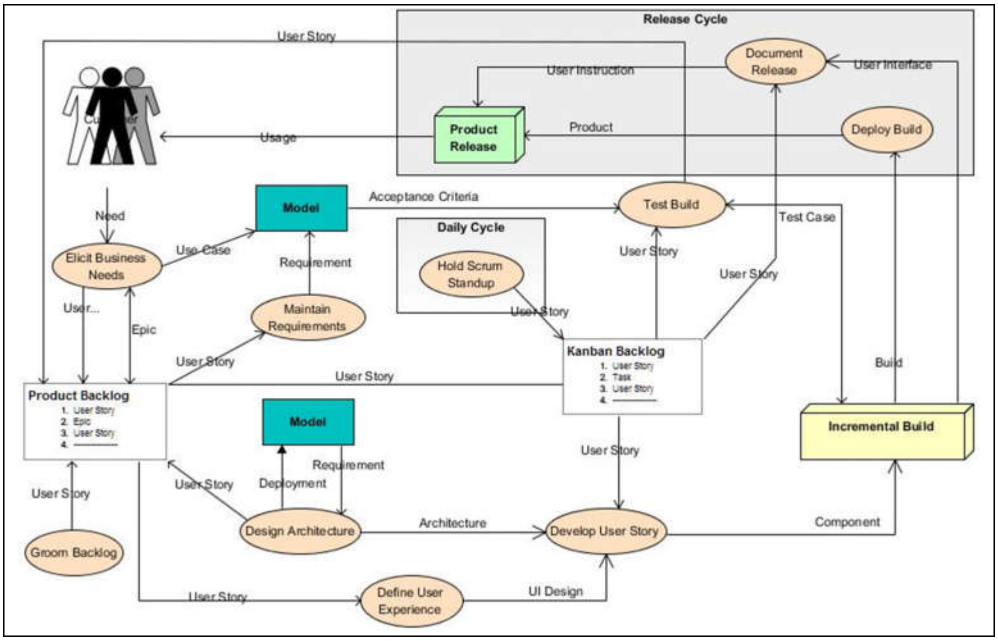

Figure 10 – Quality Through Kanban Process

Kanban is not considered in the descriptions of the process details.